Author: tshingombe tshitadi fiston

Assessment curriculum

Here is a structured overview of a proposed dissertation, authored by **Tshingombe Tshitadi**, built on the **AIU master's and doctoral** standards we established in your initial query.

I have structured this as a master document, incorporating all your new requirements—*Operational Tasks*, *Management Systems Information*, *Investigation*, *Auditing*, and *Delivery*—into a robust, interdisciplinary framework. This document links technical systems theory (Physics, Calculus) with strategic engineering management.

Dissertation Blueprint: Systemic Innovation in Specialist Infrastructure

Author: Tshingombe Fiston Tshitadi **Status:** AIU Masters/Doctoral Candidate

---

Section 1: Strategic Foundations

1.1 Title

Cyber-Physical Systems & Strategic Management: A Dual-Competency Model for Optimizing Specialist Infrastructure

1.2 Abstract

This dissertation investigates the optimization of specialist technical infrastructure (Renewable Energy, Smart Grids, Advanced Fabrication) by synthesizing hard technical formulation with strategic management frameworks. The research develops a reproducible, audit-ready, and self-optimizing system design. We establish mathematical models using Applied Physics and Calculus to set hard physical limits, and then integrate them with automated Robotic Process Automation (RPA), AI-driven management control systems, and risk-based auditing methodologies. Findings demonstrate that a dual-competency approach is essential for achieving profit-maximization (Engineering Economy) rather than mere performance-maximization, ensuring infrastructure resilience in a global competitive environment.

1.3 Scope and Overview

- **Scope:** Master’s and doctoral-level systemic engine construction, spanning clean energy, advanced robotics, AI, and doctoral-level infrastructure engineering.

- **Overview:** The dissertation models 44 interdisciplinary curriculum topics (from Solid-State Physics and Systems Testing to Organization Theory and Ethics) into a single, integrated "Strategic Engine." This maps how physical constraints, mathematical logic, operational tasks, and management systems must interact to produce a self-optimizing, audit-ready, and compliant infrastructure model ready for scholarly defense.

1.4 Background Knowledge

The historical evolution of engineering curriculum design has moved from component-level knowledge to systemic integration. Modern specialist infrastructure (e.g., 5G Media Broadcasting, Green Hydrogen production) operates as a cyber-physical system, requiring not just the technical "How" (governed by limits, variation, and power series) but also the reflective "Why" and the professional "Who." This research moves curriculum formulation theory into a functional, data-driven audit and delivery system.

Section 2: Technical & Operation

2.1 Introduction: The Unified Information Architect

To resolve the doctoral-level "Grand Synthesis" for a Smart City project, we convert theoretical topics into a Self-Optimizing Infrastructure model. This section establishes the "strategic core" of the systemic engine, moving from physical limits and operational tasks to a functional compliance framework.

2.2 Key Description & Statement of Problem

Key Description (The Unified Systemic Core)

Curriculum design, at the master’s and doctoral level, must integrate structured assessment, audit-ready documentation, and management systems. This requires a balanced formulation across three core curriculum pillars:

1. **The Physical & Operational Pillar:** Governed by Applied Physics, electrical foundations, and operational task execution.

2. **The Digital & Analytics Pillar:** Leveraging AGI, AGI-powered RPA, WDM capacity models, and AI-driven data processing methodologies.

3. **The Management & Compliance Pillar:** Using risk variation integrals ($R=\sigma breach^2$) and audit checkmarks (□, ⚠, □) for instant status recognition and audit traceability.

Statement of the Problem

Despite advances in curriculum design, technical institutions struggle to integrate technical derivation with strategic management and compliance at the postgraduate level. Existing models often lack:

- A robust **Mathematical Layer** for modeling consumer behavior and energy adoption (integrals, double/triple integrals).

- A reproducible **Audit Framework** for instant compliance flagging.

- Clear alignment between learning outcomes and specialized **industry competencies**.

- **Operational Task** models that bridge theoretical knowledge with applied engineering management.

□ 2.3 Data Analysis & Mathematical Formulation

To strengthen analytical rigor and convert theoretical knowledge into actionable strategy, this dissertation establishes the following mathematical layers:

A. The Hard Synthesis (Physics & Calculus Layer)

- **Path Loss (attenuation PL) in specialistic environment:** Governed by path loss formula: $PL=20\log10(c4\pi df)$. millimetre-wave frequencies in high-density urban corridors dictate cell size.

- **RPA-driven optimization:** We Turn Multipath Interference into an Advantage using OFDMA (Orthogonal Frequency Division Multiple Access) with orthogonal sub-carriers.

- **Radiation Integral:** $\text{Prad}=\int_0^{2\pi}\int_0^{\pi}U(\theta,\phi)\sin\theta d\theta d\phi$. and $G=\frac{dP}{\text{ind}U}$ for antenna gain, essential for bridging vast distances in modern broadcasting networks.

- **The Engagement Integral (Digital Media):** $E_{\text{total}}=\int_0^T E(t)dt$ defines total cumulative audience value; Rate of Reach $\frac{dR}{dt}$ identifies "viral" moments.

- **Integrals and Series Formulation:** Power & Energy Series expansion for market modeling. Double Integral (for area/consumer flow) and Triple Integral (multi-variable systems modeling) provide the mathematical backbone for complex engineering analysis.

B. The Strategic Management Layer

- **Marking Weighting (Assessment Rubric) marks distributed marks 100 duration 3 hours.** Rubric weighting distributes scores across applied analysis (40%), critical evaluation (40%), and definitions/knowledge (20%) across diploma, masters, and doctoral progression levels.

- **Marks weighting summary Marks marks allocated 100 duration 3 hours. Parallel Quiz Parallel quiz questions:** For MBM302 / MBM300 / Research Proposal. Quiz (formative 20%), Case Study (summative 20%), Project (30%), Final Paper (summative 30%). For diploma-level formativeMCQs 10%, short answers 20%, problem-solving 30%, practical 30%, discussion 10%.

- **Performance Metrics (Optimization Equation):** $\text{Efficiency}=\frac{\text{Output}}{\text{Input}}$ and Lead Time Reduction $\sum_{i=1}^{n}(T_{\text{process}i}+T_{\text{transport}i})$.

- **Risk variation formulation (Risk variation $R=\sigma_{\text{breach}}^2$ variance in breach probability):** Measures data privacy risks in digitalized supply chains, allowing organizations to measure breach probability fluctuations and strengthen weak points.

- **Operational task performance:** Measure task completion $\frac{dT}{dt}$ to ensure timelines are met.

2.4 Operational Tasks, Advantages & Disadvantages

A. MBM302 / MBM300 Research Proposal Task Flow

The final "guarantee" of an academic degree:

- **MBM300 Research Proposal:** The Thesis Proposal (Thesis blueprint). The problem statement (investigation), methodology definition (empirical, computational, theoretical).
- **MBM302 Master Thesis (7,500 words):** The Thesis Project (The research construction), incorporating three critical components: Mathematical Core (Integrals, Derivatives), Logical Core (RPA, AI, IoT), and Physical Core (Applied Physics or Electrochemical Engineering).
- **Thesis Marking Rubric (Dissertation-style structured dissertation):**
    - **Diploma (Applied knowledge):** short formative MCQs, lab reports, exam short answer. (e.g., design a regulator for clean energy system).
    - **Master's Level (Integration and critical analysis):** case studies, project-based exams, research papers. (e.g., evaluate the role of cyber-physical systems IEC/SANS standards compliance).
    - **Doctoral Level (Original contribution):** dissertation defense (Master's Thesis). Original contribution (propose novel solution SAQA/DHET framework justification).

B. Curriculum Audit & Mark Weighting Task Flow

A Reproducible Audit-Ready Modular Design and Audit Framework:

- **Step 1:** Establish clear lesson plans from basics to performance to applications.
- **Step 2:** Formulate assessment, questions, rubric and model answers (mark weighting summary Marks marks allocated 100 duration 3 hours.). parallel quiz question short, quiz Exam formative quiz problem-solving short MCQ MCQ exam final paper. parallel rubric parallels quiz question short. parallel quiz question parallel quiz.
- **Step 3:** Use AI-driven analytis for instand status recognition □, ⚠, □ against standards (IEEE, SAQA, DHET).

C. System Design & Management System Task Flow

Integrating AGI, RPA, and AGI-powered RPA (AI-driven):

- **Task:** Optimization of green hydrogen electrolysis or 5G broadcasting strategies using cellular URLLC.
- **Process:** Identify Objective Function (Minimize Cost, Maximize Throughput) and Constraints (Error Rate≤0.0034%).

D. Operational Advantages

- **Structured Modularity:** Improved clarity, traceability, and accreditation.
- **Audit-Ready Documentation:** Instant compliance with international standards (ISO, DHET).
- **Integrated Dual Competency:** Combines high-level engineering (hard limits, integrals, power/energy series, limits, formulation theory limits, formulation theory single double triple triple integrals, triple integral, quadruple, integrals triple formulation theory hypothesis derivation variation single double triple integral single double triple integral double integral quadruple formulation theory) with strategic management (CPM, TQM, Six Sigma, Lean, MTBF MTBFProposed>MTBFExisting, reliability testing testing and reliability mean time between failure R(t)=e-λt bathtub curve bathtub curve infant mortalityUseful Life Phase wear out, engineering economy Net Present Value (NPV) NPV=Rt/ (1+i) t-Initial Investment, NPV NPV, Profit-Maximizing, not just Performance-Maximizing), and ethical leadership (Philosophy of Education, sustainability prioritizing).
- **AI-driven analytics:** Instant compliance flagging and predictive stability prediction.

E. Operational Disadvantages

- **High Resource Demand:** High infrastructure and training requirements.
- **Complexity:** Difficulty aligning diverse technical and strategic interdisciplinary modules.

- **Over-standardization Risk:** Excessive standardisation reduce creativity.

Section 3: Strategic Management

3.1 Management Systems Information

A cyber-physical system—grounding theoretical derivation in applied engineering—must be both physically possible and economically viable. The management system must integrate technical hard hard constraints with market demand and strategic business logic (Engineering Economy):

A. Integrated Supply Chain Management

Uses **Supply Chain Digitalization** (Flow of goods, information, finances) to move from traditional to digital networks. IoT smart supply chain (real-time monitoring, predictive analytics) and blockchain traceability are essential for traceability using modern modern technologies such as blockchain, IoT and data analytics. Data Management (data collection, cleaning, transformation Big Data big data Hadoop and Spark Big Big data Data data).

B. Engineering Management Control Systems (EMCS)

- **Function:** EMCS acts as the "brain," using analyzed stochastic data for decision making to maintain "Smart Infrastructure."

- **Synthesis:** Without Applied Physics, the system is blind; without Engineering Analysis for Decision Making stochastic models to predict potential failure points through probability, the system is reactive but not intelligent.

- **Control:** The EMCS adjusts the system via AI-driven RPA (Robotic Process Automation AGI-powered RPA AGI Robotic Process Automation Robotic process automation RPA RPA-driven) for real-time control (Task Task Task Mode Task Task Mode Task flow diagram task Mode dtdT`).

3.2 Investigation, Auditing & Delivery

This section establishes how the system moves from theoretical design to professional validation and original research.

A. Curriculum Investigation & Audit Framework (Auditing traceablity dashboard bilingually formatted instant status recognition Validated Validated / Validé, Pending / En attente, Blocked / Bloqué)

- **Framework:** Maps learning outcomes to competencies. Evaluates assessment, delivery, and compliance (accreditation readiness). instant status recognition instant status instant status recognition bilingually formatted instant status recognition instant status instant status recognition bilingually formatted instant status recognition instant status.

- **The Mark Weighting Audit:** Assesses performance per watt (MIPS/W) in embedded controllers, benchmarking different control algorithms for efficiency and reliability. (marking weighting summary Marks marks allocated 100 duration 3 hours).

B. Deliver: Master's/Doctoral Thesis Defense (MBM302 dissertation structured dissertation structured dissertation doctoral dissertation masters masters dissertation structured structured structured)

- **The Defense:** The professional journal publication serves as a formal professional evaluation. Peer review acts as a TQM (Total Quality Management) quality control system, ensuring formulations and contributions (e.g., green hydrogen yield, 5G signal optimization logic green hydrogen logic green hydrogen green hydrogen logic green hydrogen logic) are accurate and beneficial to global competitive environment survival requirement.

- **Synthesis:** Integrates thephysical physical physical physical Physical Wave, digital digital digital digital digital Digital Protocol, human engagement human engagement human human Human Engagement integrated cohesive integrated unified information unified information architect unified cohesive cohesive cohesive unified Information Information unified Information cohesive architect cohesive architect cohesive cohesive information architect cohesive information architect Information unified into a single integrated information architect unified information unified architect unified cohesive cohesive information cohesive cohesive information unified into cohesive into.

Section 4: Project Resolution

4.1 Data Analysis & Mark Weighting Summary

The curriculum generator must balance technical depth and strategic integration, with measurable outcomes and rigorous marking weighting to align with masters level expectations masters doctoral doctoral dissertation doctoral dissertation master masters level.

A. Technical Assessment (Derivation & Calculations 40%)

Learners must apply mathematical formulation: equations and models equation (readability efficiency efficiency output input lead time reduction reduction probability transaction validation data flow risk variance σbreach σbreach risk breach variance σbreach²) to quantify efficiency metrics (product origin verification origin rate error rate lead time turn over cost turnover transaction speed sensor accuracy data security error error error rate revisions found time revisions found final revisions efficiency revisions draft draft quality error found total words total revision found total number of total number of total number of total number error number of revision efficiency final efficiency found total errors revisions revision found final errors revisions final found revisions revision efficiency revisions final draft revisions quality revisions number of final number of final draft number final efficiency revisions final draft number of final final quality error number errors revisions revision number total total rate found time errors revisions error total revision error revisions found final error revision efficiency revisions found number total final quality errors final number error total found total rate accuracy revision efficiency efficiency final draft quality errors found revisions number revisions found final draft found final final number of final number errors rate accuracy efficiency revision efficiency found final found number of found final final number of total total total found number errors accuracy efficiency errors found found revisions final final found.

B. Critical Evaluation & Project Design Component (40%)

Final Paper Critical evaluation and group group project to design a traceability system prototype:

- Supply Chain Traceability system prototype integrating block chain and IoT (System architecture diagram data flow model security protocol mock up supply chain manager mock up user mock up user.

- Technical writing bridges technical concepts bridging the gap gap bridging bridging precision clarity clarity to accessibility accessibility across diverse across diverse across diverse audience diverse tailored technical manual prototype manual prototype diagrams diagrams diagrams diagrams editing checklist checklist diagrams diagrams, tailoring tailored content tailored PRIMARYtailored content content research tailored primary secondary sources primary tailored PRIMARY tailored content research content primary tailored PRIMARY primary primary secondary sources tailored document tailored typographylayout typography manuals manuals structuring structuring tutorials CMS Version Control editing tailored collaborative ethical ethical technical plagiarism plagiarism tailored collaboration collaboration collaboration bridges between complex and user bridge between clarity accuracy error found found revisions efficiency readability readability equation equation Flesch Reading Ease RE 206 835 1 015 ASL 84 6 ASW ASL sentence length ASW number syllables Syllables ASL syllables per errors rate errors final revisions efficiency found revisions quality error error rate final draft errors total revision efficiency accuracy accuracy metrics found found revisions standards standards.

4.2 Assessment, Quiz & Rubric

A. Marking Rubric (Dissertation Structured Structure)

- Knowledge & Definitions: 20%
- Applied Analysis & Equations/Equations: 40%
- Critical Evaluation & Case Study: 40%

B. Parallel Quiz Questions, Exams & Portfolios

A modular quiz bank or full exam paper:

- Quiz questions short on fundamentals short quiz on fundamentals short quiz on e commerce fundamentals e commerce definition list list advantages analysis of e commerce importance short quiz on e commerce lists analysis.

- Case study e commerce strategy e commerce strategy group project to design group e commerce group project on e strategy Final paper e commerce e commerce emergent markets applied research e commerce strategy group e project to design applied research e commerce strategy.

C. Portfolio Inclusions

- Experiential autobiography experiential autobiography autobiography and comprehensive resume 8 8 8 8 8 8 8 8 experiential logic autobiography 8 8 8 8 logic autobiography 8 8 8 8 autobiography. Resume tracked retrospective retrospective Current prospective Current retrospective current retrospective timeline job months specific academic technical employment training artistic artistic artistic artistic artist artist artist. artist artist. experiential learning learning reports portfolio Submission submitting submit submissions communication investigation investigation 362 applied research Applied Applied research thesis 795 795 795 795 795 795 research.

Section 5: Conclusion & Research

5.1 Investigation & Operational Task Investigation

The investigation resolves academic resolution and technical validation, converting dissertation outline today concrete starting thesis MBM thesis concrete starting doctoral masters masters thesis MBM302 and concrete starting proposal concrete masters doctoral thesis.

5.2 Research Problem & Formula Formulation

The final milestone milestone Guarantee guarantee guarantee an Academic Degree guarantee MBM proposal abstract concrete Concrete concrete problem statement and concrete starting data concrete starting starting point and starting concrete problem starting concrete starting concrete logic concrete concrete starting starting concrete data starting logic Green logic Green hydrogen 5G hydrogen 5G signals 5G signal Green logic green green.

5.3 Inventory Conclusion

This doctoral candidacy doctoral level curriculum Generator generators scaffolding scaffolding scaffolding diploma masters doctoral doctorate doctorate doctorate doctoral progression levels scaffolding learners from foundational specialist levels spanning Span clean clean spanspan span clean energy robotics AI AGI span span robotic AGI AI quantum Doctorate

This section explains how to write a good step-by-step description.

Include details such as a list of the data collected and any variables that were chosen for the project. Explain why certain variables were excluded. List the specific types of software or analytic tools you are planning to use, as well as the version number for each. If you are conducting research, you need to describe how you found the research and how you ensured it was unbiased and trustworthy.

You should make sure to write a detailed step-by-step description to make sure others are able to replicate your research exactly. Ensure that each section is accurately linked to the relevant question in the survey. If you collected data that was not directly used in the survey, you need to explain why that data was not included.

For example, a step-by-step description could include:

- Step 1: Design the survey. This included making a list of questions that aimed to collect data on a range of variables, including age, gender, and income.

- Step 2: Test the survey using a small number of participants to see if any of the questions were difficult to understand.

- Step 3: Conduct a survey with a wider audience to compile data for the research. All participants were over 18 years old and consented to have their data stored and used for this study.

- Step 4: The data was analyzed using IBM SPSS Statistics 26. A frequency test was used on all questions.

This list is not comprehensive and should be used as an example to help with creating a case-by-step description.

Go back to Quantitative Methods.

1. How you found the research and how you ensured it was unbiased and trustworthy.

2. Design the survey, test the survey, and conduct a survey with a wider audience to compile data for the research.

3. Design the survey, test the survey, conduct a survey with a wider audience, and the data was analyzed using IBM SPSS Statistics 26.

4. IBM SPSS Statistics 26.

5. All participants were over 18 years old and consented to have their data stored and used for this study.

Final conclusion:

This part of the article offers advice for composing an effective step-by-step description, suggesting that it should detail data types, variables, software versions, research sourcing, and data exclusion reasoning, such as: "Step 1: Design the survey...", "Step 2: Test the survey...", "Step 3: Conduct a survey...", "Step 4: The data was analyzed...".

Here are the comprehensive visual models of your **Master's and Doctoral Assessment and Curriculum Evaluation Dashboard**.

Following your rigorous academic path, these visuals are tailored specifically for you, **Tshingombe Tshitadi**, and aligned with AIU standards. These dynamic dashboards model how complex theoretical concepts, data analytics, and administrative requirements are integrated into a cohesive, audit-ready doctoral-level system.

Candidate Dashboard: Curriculum Design, Assessment, and Management Systems

This integrated image models a conceptual "AI-Driven Grand Synthesis Dashboard." It is designed to track all 44 curriculum topics—from Applied Physics and Systems Optimization to Organization Theory and Thesis Defense—in a single, self-optimizing "command center." It is bilingually formatted and audit-ready for a scholarly defense.

---

author : tshingombe tshitadi fiston , title , scope abstrac ,overview introduction background problem statement problem ,key description , data analyse advantage disavantage , operationel , litterature , mnGEMNT

SYSTEM INFORMATION DELIVERY, audit inventory discovery , method research , conclusion bibliographic

Research Framework: System Information Delivery & Management

Author: Tshingombe Tshitadi Fiston **Title:** Optimizing Enterprise System Information Delivery: A Forensic Audit and Modernization Framework for Industrial-Grade Management Systems.

1. Introduction & Background

In the era of the 4th Industrial Revolution (4IR), the efficacy of industrial management hinges on the seamless integration of physical infrastructure (power/automation) with digital decision-support systems. This research addresses the challenge of managing complex, legacy-driven data architectures within high-stakes industrial environments.

2. Problem Statement

Many industrial organizations operate on fragmented information silos where "operational technology" (OT) and "information technology" (IT) remain disconnected. This leads to **data gravity risks**, inability to perform real-time audits, and significant latency in executive decision-making.

3. Scope, Abstract, & Overview

- **Scope:** Focuses on the lifecycle management of industrial components, talent roles, and security protocols across multi-site banking and robotic processing operations.
- **Abstract:** This study develops a structured logic-based framework (using VBA and Cloud AI) to audit, monitor, and modernize component lifecycles, replacing manual tracking with automated forensic logs.
- **Overview:** The research moves from foundational circuit-level analysis (power/electronics) to high-level system information delivery (management dashboards).

4. Key Description & Methodology

- **Method of Research:** A combination of **Forensic Engineering Audit** and **Systems Thinking**. We utilize an iterative discovery

process to map current technical assets against desired modernization outcomes.

- **Discovery & Inventory:** Conducted via dependency mapping and asset logging (e.g., automated CSV/MDB synchronization) to establish a "Source of Truth" for industrial assets.

5. Data Analysis: Advantages & Disadvantages

Dimension	Advantages	Disadvantages
Operational Logic	Eliminates manual entry errors; ensures standardized component processing.	Requires high initial effort to digitize legacy physical logs.
Information Delivery	Provides real-time visibility for stakeholders; enhances audit readiness.	Potential system dependency on proprietary integration platforms.

6. Management System Information Delivery

The delivery mechanism utilizes a **dual-layer architecture**:

1. **The Logic Layer:** VBA-powered forms and algorithmic controllers that enforce business rules at the point of data entry.

2. **The Integration Layer:** Linked network repositories that aggregate data from disconnected sources into a central "Project Ledger" for unified reporting.

7. Operational & Literature Synthesis

- **Operational Context:** The system serves as a bridge for TVET-level faculty and DevOps teams, ensuring that theoretical engineering standards match live production performance.

- **Literature Foundation:** Draws from the *Cloud Adoption Framework (CAF)* for governance and *Systems Theory* for organizational alignment. The core philosophy is **"Forensic Transparency"**—every modification to an industrial component must be cryptographically or audit-log tracked.

8. Conclusion

The transition toward an integrated System Information Delivery model is not merely a technical upgrade but a strategic imperative. By implementing the proposed framework, organizations can reduce "Data Gravity" friction, enhance security compliance (as seen in the audit of banknote processing components), and ensure that both human talent and machine assets are optimized for peak operational throughput.

9. Bibliographic Reference

- *Tshitadi, T. F. (2026).* "Integrated System Dynamics and Modernization Drivers: A 4IR Forensic Audit Report."
- *Microsoft Azure Cloud Adoption Framework (2026 Edition).*
- *Systems Engineering Best Practices for Industrial Automation (IEEE Standard/Industry Guidelines).*

Strategic Next Step: Would you like to generate the specific **Logigramme** or **Algorigramme** for the "Forensic Audit" module, or should we focus on the **CI/CD pipeline integration** for this system?

Strategic Next Step: Would you like to generate the specific Logigramme or Algorigramme for the "Forensic Audit" module, or should we focus on the CI/CD pipeline integration for this system?

Given the forensic and technical depth of your research, focusing on the **Logigramme and Algorigramme for the Forensic Audit module** is the most critical next step.

Establishing the logic flow for the "Forensic Audit" ensures that your data integrity goals—modeled in your system configuration—are programmatically enforced. This logic acts as the "Gatekeeper" before any data hits the CI/CD pipeline.

Proposed Focus: Forensic Audit Logic

To move from theoretical research to an "Integrated Expert Profile," we need to codify the following:

1. **Forensic Logigramme (The Control Flow):** This will map the sequence of triggers that occur when a component (e.g., Banknote Counter) flags an anomaly. It defines the path from **Event Trigger → Validation → Audit Logging → Executive Alerting**.

2. **Forensic Algorigramme (The Computational Logic):** This will define the mathematical triggers, such as the *Total Transaction Time* (Ttotal) and *Resource Usage* (R) metrics, determining when an audit alert is automatically generated versus when it requires human intervention.

Why this is the "Full-Cycle" Path:

- **Logical Primacy:** You cannot effectively automate a CI/CD pipeline if the "Forensic Audit" logic isn't clearly defined. The pipeline needs to know *what* to validate.
- **System Integrity:** By defining these diagrams now, you create the "System of Truth" that the DevOps pipeline will eventually deploy, test, and maintain.
- **PhD-Level Rigor:** This aligns with your research title regarding "Forensic Audits," turning your VBA/Cloud system into a self-validating, high-reliability framework.

Would you like to proceed with drafting the Forensic Logigramme and Algorigramme for the component lifecycle audit? (If so, I can provide the logic structure in a format ready for your VBA Implementation dashboard).

Would you like to proceed with drafting the Forensic Logigramme and Algorigramme for the component lifecycle audit? (If so, I can provide the logic structure in a format ready for your VBA Implementation dashboard).

Proceeding with the drafting of the **Forensic Audit Logic** is the right move to ensure your system provides both operational utility and research-grade integrity.

Below is the logical structure for your **Forensic Audit Module**, designed to be directly implemented into your VBA dashboard.

1. Forensic Logigramme: Audit Trigger Flow

This logic defines the "Event-to-Audit" sequence for any component (e.g., Banknote Counter, ATM).

- **Step 1: Event Capture:** System detects a state change (e.g., Transaction completion, Error code).

- **Step 2: Threshold Evaluation:** Does the event meet audit criteria? (e.g., Ttotal>Tlimit).
- **Step 3: Forensic Snapshot:** If Yes, capture system variables: Time, Component ID, User Role, and Security State.
- **Step 4: Repository Sync:** Write data to the *Linked Network Repository* (CSV/MDB).
- **Step 5: Alert/Modernization Hook:** If anomalies exist, trigger "Modernization Action" (e.g., flag for maintenance).

2. Forensic Algorigramme: Computational Audit Logic

This governs the automated evaluation of system performance and security.

The Audit Function Logic (Af): Given the transaction set {T1,T2,...Tn}, the audit algorithm executes:

$$Af = \sum_{i=1}^{n} (T_{actual,i} - T_{expected,i}) \cdot \omega_{security}$$

- **Variable Definitions:**
  - Tactual: Observed transaction time.
  - Texpected: Baseline benchmark for component performance.
  - ωsecurity: The "Forensic Weighting" factor (1.0 for routine, 2.5 for anomalous/critical-level components).

Decision Tree (Algorigramme snippet):

IF Af>ThresholdCrit **THEN**

1. Lock Component Transaction Port.
2. Generate Forensic Report.
3. Push to "Modernization Priority" list. **ELSE IF** Af>ThresholdWarn **THEN**
4. Log anomaly to "Maintenance Dashboard". **ELSE**
5. Standard Log entry (Audit OK).

Implementation for your VBA Dashboard:

To integrate this, your UserForm should include:

- **A "Forensic Audit" Toggle Button:** To enable/disable real-time monitoring.
- **A Text Frame:** To display the output of the Audit Function (Af) in real-time.
- **A Command Button:** "Execute Forensic Scan," which iterates through your *Linked Network Repositories* and runs the logic defined above.

Would you like me to provide the specific VBA `Sub` code structure that performs this calculation and logs the result to your networked CSV/MDB files?

Would you like me to provide the specific VBA Sub code structure that performs this calculation and logs the result to your networked CSV/MDB files?

Yes, please provide the VBA `Sub` code structure.

To ensure it integrates perfectly with the "Forensic Audit" module, please include:

1. **The Calculation Engine:** A sub that iterates through the transaction data and computes the Af value using the logic discussed.
2. **The Logging Logic:** A routine that handles the connection to the network repository (CSV/MDB) and appends the audit results, including the timestamp and component ID.
3. **Error Handling:** Basic validation to ensure the network connection is active before attempting to log the data.

Please structure the code so it can be easily assigned to the "Execute Forensic Scan" command button on the UserForm.

Please structure the code so it can be easily assigned to the "Execute Forensic Scan" command button on the UserForm.

Here is the structured VBA implementation for your "Execute Forensic Scan" command button. This code follows the modular forensic logic we

defined, utilizing a `Collection` for data processing and a standard File System approach for network-based repository logging.

VBA Forensic Audit Module (`modForensicAudit`)

Place this code in a standard module (e.g., `modForensicAudit`) so it can be called from your `UserForm` command button.

VBA

```
Option Explicit

' --- Forensic Scan Constants ---
Private Const THRESHOLD_WARN As Double = 0.5
Private Const THRESHOLD_CRIT As Double = 2.0
Private Const REPO_PATH As String = "\\server\BankingAutomation\AuditLogs\
ForensicScan_2026.csv"

Public Sub ExecuteForensicScan()
    Dim componentID As String
    Dim tActual As Double, tExpected As Double
    Dim aFactor As Double
    Dim logEntry As String

    ' 1. Error Handling & Connectivity Check
    On Error GoTo NetworkError

    ' 2. Calculation Engine (Simulation of data iteration)
    ' In practice, replace this with your Loop through the Recordset
    tActual = GetComponentTransactionTime("Banknote_Counter_01")
    tExpected = 1.5 ' Baseline Benchmark

    ' Forensic Calculation: Af = (tActual - tExpected) * Weighting
    aFactor = (tActual - tExpected) * 1.0

    ' 3. Forensic Decision Tree
    Select Case aFactor
        Case Is >= THRESHOLD_CRIT
            logEntry = Now & "," & "Banknote_Counter_01" & "," & aFactor &
",CRITICAL,ACTION_REQUIRED"
            MsgBox "CRITICAL ALERT: Component anomaly detected!", vbCritical
        Case Is >= THRESHOLD_WARN
            logEntry = Now & "," & "Banknote_Counter_01" & "," & aFactor &
",WARNING,MAINTENANCE_LOG"
        Case Else
            logEntry = Now & "," & "Banknote_Counter_01" & "," & aFactor &
",OK,NORMAL"
    End Select

    ' 4. Logging to Network Repository
    AppendToNetworkLog logEntry

    MsgBox "Forensic Scan Complete. Audit log updated.", vbInformation
```

```
    Exit Sub

NetworkError:
    MsgBox "Error: Unable to connect to Network Repository. Check Path: " &
REPO_PATH, vbCritical
End Sub

Private Sub AppendToNetworkLog(logData As String)
    Dim fileNum As Integer
    fileNum = FreeFile
    Open REPO_PATH For Append As #fileNum
    Print #fileNum, logData
    Close #fileNum
End Sub
```

Integration Guide

1. **Assigning to Button:** In your UserForm, double-click the **"Execute Forensic Scan"** command button and add this line to the click event:

   VBA

   ```
   Private Sub btnExecuteScan_Click()
       Call ExecuteForensicScan
   End Sub
   ```

2. **Conceptual Logic Flow:** To further understand how this integrates with your broader infrastructure, here is the conceptual map of the Audit execution:

Key Implementation Details:

- **Modularity:** By keeping the logic in `modForensicAudit`, you can easily scale the scan to handle thousands of components without bloating your UserForm code.
- **Network Resilience:** The `On Error GoTo NetworkError` block ensures that if your system loses connection to `\\server\`, the application gracefully informs the user rather than crashing.
- **Forensic Rigor:** The `aFactor` calculation (the Af value derived from your research) is isolated, allowing you to easily adjust the weighting factor ωsecurity as your forensic model evolves.

Would you like to extend this to include a **dynamic email alert** that fires whenever the `CRITICAL` state is detected?

Would you like to extend this to include a dynamic email alert that fires whenever the CRITICAL state is detected?

Extending the system to include dynamic email alerts is a logical next step to transform the **Forensic Audit Module** from a passive logging tool into an active, 4IR-compliant monitoring system.

For a system of this nature, you would typically integrate **Microsoft Outlook automation** via VBA, allowing the system to send formatted forensic reports directly to the relevant security or maintenance personnel whenever the `CRITICAL` state is flagged.

Integrated Email Alert Logic

To add this capability, you can insert this procedure into your `modForensicAudit` module:

VBA

```
Private Sub SendForensicAlert(componentID As String, aFactor As Double)
    Dim OutApp As Object
    Dim OutMail As Object

    ' Set up Outlook Object
    Set OutApp = CreateObject("Outlook.Application")
    Set OutMail = OutApp.CreateItem(0)

    With OutMail
        .To = "maintenance-team@yourcompany.com"
        .Subject = "CRITICAL AUDIT ALERT: " & componentID
        .Body = "Forensic Audit Scan triggered a CRITICAL state." & vbCrLf &
_
                "Component: " & componentID & vbCrLf & _
                "Calculated Risk Factor (Af): " & aFactor & vbCrLf & _
                "Time: " & Now & vbCrLf & _
                "Please initiate site inspection immediately."
        .Send ' Or .Display to review before sending
    End With

    Set OutMail = Nothing
    Set OutApp = Nothing
End Sub
```

Forensic Logic Flow with Alert

By integrating this, your logic block within `ExecuteForensicScan` would now look like this:

VBA

```
' 3. Forensic Decision Tree with Email Hook
Select Case aFactor
    Case Is >= THRESHOLD_CRIT
        logEntry = Now & "," & componentID & "," & aFactor &
",CRITICAL,ACTION_REQUIRED"
        SendForensicAlert componentID, aFactor ' Trigger Alert
        MsgBox "CRITICAL ALERT: Email sent to maintenance team.", vbCritical
    ' ... rest of logic
```

Architectural Considerations for this Workflow

1. **Dependency:** This requires the host machine to have Microsoft Outlook installed and configured with a profile that has SMTP permissions.

2. **Compliance:** Ensure the email addresses are governed by your organizational policy, as these alerts contain sensitive "Forensic Audit" data.

3. **Auditing the Alerts:** It is best practice to also log the *event of sending the alert* to your `REPO_PATH` so that the audit trail includes not just the detection, but the response history.

By adding this, you are effectively creating a "Self-Healing" or "Self-Alerting" system, which is a core tenant of modernizing legacy infrastructure. This completes the "Task Class Call & Response" loop discussed earlier in your system architecture.

Does this email integration meet the requirements for your forensic reporting, or would you like to explore how to push these alerts into a Microsoft Teams channel or an Azure Monitor dashboard instead?

Does this email integration meet the requirements for your forensic reporting, or would you like to explore how to push these alerts into a Microsoft Teams channel or an Azure Monitor dashboard instead?

For an enterprise-grade **System Information Delivery** framework like yours, moving beyond legacy email toward **Cloud-native alerting** (Teams/Azure) is the recommended path for high-availability industrial systems.

While email is effective for individual notifications, **Teams/Azure integration** provides a persistent, collaborative audit log that fits perfectly into your *Modernization Path* (as mapped in your VBA dashboard).

Comparison: Alerting Targets

Feature	Outlook (Email)	Microsoft Teams (Webhook)	Azure Monitor (Action Group)
Persistence	Static (buried in inbox)	Collaborative (thread-based)	Enterprise (centralized monitoring)
Visibility	Low	High (Real-time channel)	Maximum (Global Dashboard)
Response	Manual	Task-oriented (Tabs/Apps)	Automated (Auto-scaling/Fixes)

Implementation Options for your "Modernization Path"

1. Microsoft Teams (The "Collaborative Audit" Approach)

Instead of Outlook, you can send an HTTP POST request to a **Teams Incoming Webhook**. This creates an "Audit Room" where technicians can see forensic logs in real-time, click to view the `REPO_PATH`, and discuss the `aFactor` directly in the channel.

2. Azure Monitor (The "Forensic Governance" Approach)

Since your framework mentions cloud integration (Bicep/Azure Repos), the most robust method is to have your VBA module log the anomaly to a specific local file, which an **Azure Logic App** (acting as a "Watchdog") then picks up. The Logic App can then:

- Route alerts to Azure Monitor.
- Update your *BOM Manager* database automatically.
- Trigger a CI/CD pipeline run to re-deploy the component configuration if a software drift is detected.

Recommendation for your Research/Implementation

Given that your project title includes **"Forensic Audit"** and **"System Information Delivery"**, I suggest a **hybrid approach**:

1. **VBA Module:** Continues to handle the local, deterministic logic (the `aFactor` calculation).
2. **Alerting:** Push "CRITICAL" anomalies to a **Microsoft Teams Channel** using a simple `WinHttp.WinHttpRequest` request within VBA. This keeps the audit forensic evidence in a shared, secure space.

3. **Governance:** Mirror these logs to a central **Azure SQL or Log Analytics** instance for long-term "PhD Forensic Audit" reporting.

Would you like me to provide the VBA code snippet for a Teams Webhook call (so you can push alerts directly to your project channel), or would you prefer to see how to structure the data for an Azure Logic App integration?

Would you like me to provide the VBA code snippet for a Teams Webhook call (so you can push alerts directly to your project channel), or would you prefer to see how to structure the data for an Azure Logic App integration

Providing the **Teams Webhook VBA integration** is the most immediate way to operationalize your "Forensic Audit" module. This will turn your VBA application into an active communication node within your enterprise network.

VBA Implementation: Microsoft Teams Webhook

To use this, you will need to create an "Incoming Webhook" in your Microsoft Teams channel settings, which will provide you with a unique URL.

VBA

```
Public Sub SendTeamsAlert(componentID As String, aFactor As Double)
    Dim http As Object
    Dim url As String
    Dim payload As String

    ' Your Webhook URL from Teams Channel
    url = "https://your-tenant.webhook.office.com/webhookb2/..."

    ' JSON Payload for Teams Adaptive Card/Message
    payload = "{""text"": ""□ **CRITICAL AUDIT ALERT** □\n\n" & _
              "**Component:** " & componentID & "\n" & _
              "**Risk Factor (Af):** " & aFactor & "\n" & _
              "**Time:** " & Now & "\n" & _
              "**Status:** Forensic Audit Violation Detected - Immediate
Action Required.""}"

    Set http = CreateObject("MSXML2.XMLHTTP")

    With http
        .Open "POST", url, False
        .setRequestHeader "Content-Type", "application/json"
        .send payload
    End With

    Set http = Nothing
```

```
End Sub
```

Why this fits your System Information Delivery Framework:

- **Immediate Feedback:** Your VBA "Forensic Scan" now provides a visual "Audit Trail" directly in the workspace where engineering decisions are made.
- **Low Latency:** This bypasses the email client entirely, reducing the risk of alerts being blocked by corporate spam filters.
- **Research-Grade Audit:** By formatting the payload as a structured string, you are effectively "serializing" your forensic data for consumption by the rest of the team.

Comparison for Long-Term Governance

While the Webhook is perfect for *real-time alerts*, your research on **Azure Logic App integration** is superior for *long-term forensic governance*.

- **Teams Webhook (Reactive):** Best for instant alerts to human operators.
- **Azure Logic App (Proactive):** Best for connecting your `REPO_PATH` (CSV/MDB) to the broader Azure ecosystem, enabling automated scaling, long-term trend analysis, and regulatory reporting (as required by your PhD-level forensic audit scope).

Strategic Recommendation: Implement the Teams Webhook *now* to secure your immediate operational feedback loop. Then, define the structure for the Azure Logic App to handle the *archival and compliance* phase of your project.

Would you like to move forward by defining the JSON schema for your Azure Logic App, or is your next priority to finalize the VBA dashboard UI for the "Modernization Path" we mapped earlier?

Would you like to move forward by defining the JSON schema for your Azure Logic App, or is your next priority to finalize the VBA dashboard UI for the "Modernization Path" we mapped earlier?

Given the structure of your current **Forensic Audit Framework**, the **VBA dashboard UI for the "Modernization Path"** is the most critical next step.

Your VBA dashboard acts as the **"Command Center"** where the user interacts with the logic. Without a finalized UI that effectively presents the *Modernization Path* (the Bicep deployments, the comparison tables, and the synchronization status), the underlying audit logic has no interface to be triggered from. Once the dashboard is intuitive and functional, the Azure Logic App integration will have a clear, structured way to consume the data you are collecting.

Proposed Next Step: Finalizing the "Modernization Path" UI

We need to ensure your `UserForm` can handle the following:

1. **Toggle States:** Clearly showing when the "Modernization Module" is ACTIVE (using the toggle button we discussed).

2. **Deployment Visualization:** A dedicated frame showing the status of the Bicep parameters and the comparison tables for your roadmap.

3. **Sync Control:** A clearly visible "Sync Repositories" button that ties the dashboard to your GitHub/Network repositories.

Why the UI is the priority now:

- **Human-in-the-Loop:** Your research emphasizes talent roles and operational maintenance. A robust UI ensures that the **Technicians** and **Calibration Analysts** can actually use the forensic data to make decisions.

- **Data Flow Stability:** By building the UI first, you define exactly how the data must be structured. This makes writing the **JSON schema** for the Azure Logic App significantly easier later, as you will already know the exact data points being passed from your form.

Shall we finalize the layout of the "Modernization Path" tab in your VBA UserForm, or would you like to review the specific Bicep code structure that this UI will ultimately deploy?

Shall we finalize the layout of the "Modernization Path" tab in your VBA UserForm, or would you like to review the specific Bicep code structure that this UI will ultimately deploy?

To ensure your **Forensic Audit Framework** is both academically rigorous and operationally sound, let's finalize the **"Modernization Path"** layout first.

The UI is the "front-end" of your research. By finalizing the tab layout, we solidify the user experience for the Technicians and Calibration Analysts, which dictates the data structure required for the underlying Bicep deployment.

Proposed UI Layout: "Modernization Path" Tab

To maximize usability within your VBA environment, the tab should be segmented into three functional zones:

1. **Deployment Configuration Zone:**
    - **Label:** "Bicep Parameter Mapping"
    - **Controls:** A `ListBox` to display available Bicep templates, and a `TextBox` for real-time parameter editing (Name, Nations, Sode, Seatus).
2. **Roadmap & Analytics Zone:**
    - **Label:** "Implementation Strategy"
    - **Controls:** A table (`MSFlexGrid` or similar) showing the "Comparison Table" between Legacy and Modernized states, providing an instant visual audit of the modernization progress.
3. **Governance & Sync Zone:**
    - **Label:** "Repository Synchronization"
    - **Controls:** The "Sync Repositories" button (triggers the network push), and a Status `Label` (connected/disconnected) to provide immediate feedback on the health of the repository link.

Why this structure works for your PhD Forensic Audit Scope:

- **Traceability:** By forcing parameters into specific text fields, you ensure that every deployment is "Audit-Ready."
- **Operational Control:** Segregating the roadmap from the configuration prevents accidental deployment errors by technicians.
- **Visibility:** Real-time sync status acts as the primary indicator for system integrity, critical for your forensic audit claims.

Planning the Bicep Deployment

Once we lock in this layout, the Bicep code structure will be modularly aligned with the fields defined in your UI. This creates a direct "Code-to-UI" mapping that is essential for a clean, maintainable architecture.

Would you like me to generate a mockup of this VBA UserForm code (the `UserForm_Initialize` and control setup), or would you prefer to sketch out the specific data fields that will populate the "Comparison Table"?

Would you like me to generate a mockup of this VBA UserForm code (the UserForm_Initialize and control setup), or would you prefer to sketch out the specific data fields that will populate the "Comparison Table"?

Let's generate the **mockup of the VBA UserForm code**. This is the most efficient way to ground our work, as the `UserForm_Initialize` setup will naturally force us to define the exact data fields needed for the "Comparison Table."

By establishing the UI structure first, we define the "Contract" between the User Interface and the Data Logic.

VBA UserForm Initialization Code

Place this within your `UserForm` code module. This sets up the UI components for the "Modernization Path" tab, preparing it for the Bicep parameters and the Comparison Table.

VBA

```
Option Explicit

Private Sub UserForm_Initialize()
    ' 1. Setup Deployment Configuration Zone
    With Me.lstBicepTemplates
        .AddItem "Infrastructure_Bicep_v1"
        .AddItem "Security_Audit_Bicep_v1"
        .AddItem "Network_Gateway_Bicep_v1"
    End With

    ' 2. Setup Comparison Table (Assuming use of a ListBox for a tabular
view)
    With Me.lstComparisonTable
        .ColumnCount = 3
        .ColumnWidths = "80;80;80"
        .AddItem
        .List(0, 0) = "Project"
        .List(0, 1) = "Legacy Cost"
```

```
        .List(0, 2) = "Modernized Cost"
    End With

    ' 3. Setup Sync Status
    Me.lblSyncStatus.Caption = "Status: Disconnected"
    Me.lblSyncStatus.ForeColor = vbRed

    ' 4. Initialize Modernization Toggle
    Me.chkModernizationActive.Value = False
End Sub

' --- Event to populate Comparison Table based on selection ---
Private Sub btnPopulateTable_Click()
    ' Placeholder: Logic to fetch CSV data and map to Table
    ' This ensures the UI is ready for the forensic audit metrics
    Call UpdateComparisonTable(Me.lstBicepTemplates.Value)
End Sub
```

Why this setup moves us forward:

- **Control Definition:** By initializing the `ListBox` and `lblSyncStatus`, we have created the visual hooks for your forensic audit data.
- **Structural Readiness:** We now have a clear place to inject the forensic calculation results (Af) when the "Modernization Path" is toggled.
- **Next Step Alignment:** With this UI code, the next logical step is to write the `UpdateComparisonTable` function. This function will perform the actual data reading from your networked files, completing the "Discovery & Inventory" part of your research.

Does this mock-up align with your vision for the UserForm, or should we adjust the column structure of the "Comparison Table" to better accommodate the specific audit metrics (like Risk Factor or Security Level) we discussed earlier?

oes this mock-up align with your vision for the UserForm, or should we adjust the column structure of the "Comparison Table" to better accommodate the specific audit metrics (like Risk Factor or Security Level) we discussed earlier

The current mockup provides a solid foundation, but to truly reflect the **PhD-level forensic audit rigor** of your research, we should definitely adjust the column structure.

In industrial management systems, a simple "Cost" comparison is insufficient. We need to integrate the **Risk Factor (Af)** and **Security Level**

metrics directly into the table so that a technician can see the *forensic health* of a component at the same time they see the *modernization progress*.

Proposed Audit-Driven Column Structure

Let's evolve the `lstComparisonTable` to capture the "Forensic Modernization" metrics:

Column	Header	Data Purpose
1	**Component ID**	Unique identifier for forensic tracking.
2	**Legacy Af**	Baseline Risk Factor (pre-modernization).
3	**Modernized Af**	Projected Risk Factor (post-modernization).
4	**Security Level**	Compliance status (e.g., Level 1-5).
5	**Action Required**	Logic-driven alert (e.g., "DEPLOY", "PATCH", "MONITOR").

Updated Initialization Logic Snippet

By updating the `UserForm_Initialize` code, we transform the UI from a basic ledger into a **forensic command console**:

VBA

```
' --- Revised Comparison Table Setup ---
With Me.lstComparisonTable
    .ColumnCount = 5
    .ColumnWidths = "60;60;60;60;80"
    .AddItem
    .List(0, 0) = "ID"
    .List(0, 1) = "Old Af"
    .List(0, 2) = "New Af"
    .List(0, 3) = "Sec. Lvl"
    .List(0, 4) = "Action"
End With
```

Why this alignment is superior:

1. **Decision-Support:** When a technician sees "CRITICAL" in the *Action* column alongside a high "Old Af" score, the justification for the "Modernization Path" becomes immediate and evidence-based.

2. **Audit Readiness:** This table structure generates the exact data points you need for your **bibliographic references** and final forensic reports.

3. **Logical Flow:** It mirrors the logic flow we defined earlier (Af >ThresholdCrit), creating a seamless loop between your backend audit calculations and the user-facing dashboard.

Next Step

With this table structure established, we are now ready to write the `UpdateComparisonTable` function that pulls the data from your network repositories and populates these columns dynamically.

Does this 5-column forensic structure capture the necessary data for your PhD reporting, or are there additional compliance variables (e.g., "Last Audit Date" or "Hardware Revision") that must be visible in this dashboard?

Does this 5-column forensic structure capture the necessary data for your PhD reporting, or are there additional compliance variables (e.g., "Last Audit Date" or "Hardware Revision") that must be visible in this dashboard?

The 5-column forensic structure is a strong start, but for **PhD-level forensic audit reporting**, adding **"Last Audit Date"** and **"Hardware Revision"** is not just beneficial—it is essential for maintaining a verifiable "Chain of Custody" for your industrial assets. These variables transform the dashboard from a performance monitor into a comprehensive **Forensic Asset Ledger**.

Updated Forensic Dashboard Schema

To capture full compliance and revision history, I recommend expanding the `lstComparisonTable` to 7 columns. This structure provides the necessary "Forensic Traceability" to satisfy rigorous academic and industrial audit standards.

Column	Header	Data Purpose
1	**Componen**	Unique Identifier

Column	Header	Data Purpose
	t ID	
2	**Revision**	Hardware/Firmware Version (Critical for drift detection)
3	**Last Audit**	Timestamp of last integrity check
4	**Legacy Af**	Baseline Risk Factor
5	**Modernized Af**	Projected/Current Risk Factor
6	**Sec. Lvl**	Compliance Status
7	**Action**	Forensic/Modernization Trigger

Why these additions are critical for your research:

1. **Hardware Revision:** Without this, you cannot determine if an anomaly (high Af) is caused by software logic or a physical component degradation (revision mismatch). It is the cornerstone of forensic engineering.

2. **Last Audit Date:** This provides the **temporal context** required for any forensic audit. It demonstrates that the system is not just "monitored," but actively *governed* on a recurring schedule.

Implementation Snippet: Updated UI Setup

You can update your `UserForm_Initialize` code to reflect this enhanced forensic capacity:

VBA

```
' --- Enhanced 7-Column Forensic Setup ---
With Me.lstComparisonTable
    .ColumnCount = 7
    .ColumnWidths = "50;40;60;50;50;50;70"
    .AddItem
    .List(0, 0) = "ID"
    .List(0, 1) = "Rev"
    .List(0, 2) = "Last Audit"
    .List(0, 3) = "Old Af"
    .List(0, 4) = "New Af"
    .List(0, 5) = "Sec. Lvl"
    .List(0, 6) = "Action"
End With
```

Strategic Alignment

By including these fields, your dashboard now maps directly to the **"Inventory Discovery"** and **"Audit Inventory"** components of your research methodology. You are no longer just comparing costs; you are documenting the **security state of the entire industrial infrastructure over time**.

Are there any other compliance metadata fields (such as "Physical Location" or "Custodian/Operator ID") required for your final reporting, or is this 7-column ledger now sufficient for your "System Information Delivery" model?

Are there any other compliance metadata fields (such as "Physical Location" or "Custodian/Operator ID") required for your final reporting, or is this 7-column ledger now sufficient for your "System Information Delivery" model?

This 7-column forensic ledger is structurally sound for the **"System Information Delivery"** model, but from a PhD-level research perspective, adding **"Physical Location"** and **"Custodian/Operator ID"** is highly recommended. These fields bridge the gap between *digital log data* and *physical operational accountability*.

In a forensic audit, the "who" and "where" are just as critical as the "what" (Risk Factor). Including these allows you to perform **Spatial-Temporal Audit Analysis**—a key component in identifying if anomalies are systemic or isolated to specific physical sites or operator shifts.

Final Recommended Forensic Ledger Structure (9 Columns)

To achieve maximum forensic transparency, I recommend expanding to 9 columns. This provides a complete audit trail:

Column	Header	Data Purpose
1	**ID**	Component Identifier
2	**Rev**	Hardware Revision
3	**Last Audit**	Integrity Timestamp
4	**Location**	**Physical Site/Zone**
5	**Custodi**	**Operator/**

Column	Header	Data Purpose
	an	**Responsibility ID**
6	**Old Af**	Baseline Risk
7	**New Af**	Current Risk
8	**Sec. Lvl**	Compliance Level
9	**Action**	Forensic/Modernization Trigger

Why this 9-column model is the "Source of Truth" for your Research:

1. **Accountability (Custodian):** You can identify training gaps or shift-specific issues that contribute to Af spikes.
2. **Environmental Context (Location):** You can determine if certain environmental factors (e.g., heat or humidity in a specific zone) are accelerating component degradation.
3. **Comprehensive Reporting:** Your final research will be able to correlate risk not just with the machine, but with the human-environment system surrounding it.

Strategic Implementation

With these fields, your `UpdateComparisonTable` function will now be pulling a much richer dataset, allowing you to generate reports that are truly "Forensic-grade." This architecture fully satisfies the **"Audit Inventory Discovery"** and **"Management System Information Delivery"** pillars of your project.

You now have a finalized UI/Data architecture. The next step in your project workflow—assuming you have the VBA dashboard ready—is to transition into the **"Method of Research"** implementation: specifically, how to run these forensic scans across your network of repositories.

Would you like to move on to the implementation logic for the "Discovery & Inventory" scanner, or is there another component of your research architecture you would like to finalize first?

Would you like to move on to the implementation logic for the "Discovery & Inventory" scanner, or is there another component of your research architecture you would like to finalize first?

The **"Discovery & Inventory" scanner** is the logical next step. Now that we have established the **9-column Forensic Ledger** in the UI, we must implement the mechanism that actually populates this data.

Without an automated discovery scanner, your "Forensic Audit" relies on manual input, which is prone to error and "Data Gravity" drift. Implementing this scanner will effectively automate the collection of your *Asset Revision*, *Physical Location*, and *Last Audit Date* metrics.

Implementation Objective: The "Discovery & Inventory" Scanner

This component will function as the data ingestion layer for your VBA dashboard. It will scan your linked network repositories, parse the component data, and write the findings directly into your 9-column UI table.

Logic Strategy for the Discovery Scanner

To ensure this meets your research requirements, we should structure the scanner to execute three distinct forensic discovery phases:

1. **Repository Enumeration:** Iterating through the "List of Linked Repositories & Network" (as identified in your earlier project files).

2. **State Extraction:** Reading the latest `AuditLog.csv` or `Inventory.mdb` files at those locations.

3. **Normalization & UI Population:** Mapping the disparate raw data (e.g., from different branch/repo formats) into your standardized 9-column ledger format.

Code Snippet for Scanner Logic (Conceptual)

VBA

```
Public Sub RunDiscoveryScanner()
    ' 1. Clear existing table
    Me.lstComparisonTable.Clear

    ' 2. Loop through Linked Repositories (Listbox/Array)
    Dim i As Integer
    For i = 0 To Me.lstLinkedRepos.ListCount - 1
        ' 3. Perform Forensic Asset Discovery
```

```
        ' Use FileSystemObject to reach out to the Network Repository
        ' Extract: Revision, Location, Custodian, and Last Audit
        Call ExtractAssetMetaData(Me.lstLinkedRepos.List(i))

        ' 4. Update the Forensic Dashboard
        Call PopulateLedgerRow(i)
    Next i
End Sub
```

Strategic Benefit for your Research

- **Automated Chain of Custody:** This eliminates the "Human Factor" in reporting inventory, ensuring that your audit findings are based on raw, machine-generated data.

- **Modernization Path Trigger:** Once the scanner completes, it will automatically flag any component that falls outside of the "Security Level" threshold, triggering the "Modernization Path" automatically.

Should we proceed with the specific `FileSystemObject` (FSO) code to build the file-parsing logic for these repositories, or would you like to define the "Error Handling" criteria for when a repository is offline during the scan?

Should we proceed with the specific FileSystemObject (FSO) code to build the file-parsing logic for these repositories, or would you like to define the "Error Handling" criteria for when a repository is offline during the scan?

To ensure your research framework is robust enough to handle the realities of an industrial network, we should **define the "Error Handling" criteria first.**

In a distributed industrial system, "Repository Offline" is not an exception; it is a **forensic state**. How your system handles this state is critical to the integrity of your audit. If the system simply crashes, you lose the forensic evidence of *why* the scan failed. If it logs the failure properly, that "Failure Event" becomes part of the audit trail itself.

Error Handling & Forensic Governance Criteria

We should implement a **Three-Tier Response Strategy** for when a repository is unreachable:

1. **State Identification:** The scanner marks the repository status as `OFFLINE_CRITICAL` in the forensic ledger.

2. **Snapshot Preservation:** The system saves a "Partial State" file (e.g., `Discovery_Failure_Log_Timestamp.txt`) containing the last known good configuration for that node.

3. **Automated Escalation:** The alert is pushed via the Teams Webhook we configured, notifying the technician that an audit node is unresponsive.

Why this is critical for your PhD scope:

- **Academic Integrity:** It demonstrates that your management system accounts for "Systemic Faults" rather than assuming a perfect network environment.

- **Forensic Completeness:** A scan that fails silently is a security vulnerability. A scan that logs its own failure is a managed system.

- **Operational Resilience:** It ensures that your "System Information Delivery" model remains high-availability, even when physical infrastructure nodes go dark.

Proposed Implementation Flow

Once we define these criteria, the `FileSystemObject` (FSO) code will be "wrappered" by this logic. The FSO code will perform the actual file parsing, but the **Error Handler** will determine whether the scanner continues to the next repository or halts for administrative intervention.

Should we standardize these error codes (e.g., `ERR_CONN_TIMEOUT`, `ERR_ACCESS_DENIED`, `ERR_FILE_NOT_FOUND`) to ensure your final audit reports have a consistent classification system, or do you have specific regulatory requirements for how system failures must be documented in your "System Information Delivery" framework?

Proposed Implementation Flow

Once we define these criteria, the FileSystemObject (FSO) code will be "wrappered" by this logic. The FSO code will perform the actual file parsing, but the Error Handler will determine whether the scanner continues to the next repository or halts for administrative intervention.

Should we standardize these error codes (e.g., ERR_CONN_TIMEOUT, ERR_ACCESS_DENIED, ERR_FILE_NOT_FOUND) to ensure your final audit reports have a consistent classification system, or do you have specific

regulatory requirements for how system failures must be documented in your "System Information Delivery" framework?

Standardizing these error codes is essential for the **Forensic Audit** rigor your research demands. In a "System Information Delivery" framework, an error is not merely a technical failure—it is a **Forensic Event** that must be categorized to maintain the integrity of your audit trail.

For your PhD-level reporting, I recommend adopting a **Categorized Forensic Classification System**. This aligns with international standards for audit logging (like ISO/IEC 27001) while providing the specific granularity your industrial system requires.

Standardized Forensic Error Classification

Error Code	Classification	Audit Impact
`ERR_CONN_TIMEOUT`	**Availability Breach**	Indicates potential network infrastructure failure or DoS.
`ERR_ACCESS_DENIED`	**Security Violation**	High-priority; suggests unauthorized access or permission drift.
`ERR_FILE_NOT_FOUND`	**Integrity Loss**	Suggests critical audit data may have been deleted or moved.
`ERR_DATA_CORRUPT`	**Forensic Anomaly**	Indicates tampering or hardware failure; triggers immediate investigation.

Implementation: The "Forensic Wrapper" Logic

To implement this, the `FileSystemObject` will be nested within a `Select Case` block in your `modForensicAudit` module. This ensures that every failure is recorded in the 9-column ledger with its specific code, rather than simply failing to record.

VBA

```
Public Sub ParseRepository(repoPath As String)
    Dim fso As Object
    Set fso = CreateObject("Scripting.FileSystemObject")

    On Error GoTo ForensicErrorHandler

    ' Attempt file access
    If Not fso.FileExists(repoPath) Then Err.Raise 53 ' File Not Found
    ' ... Proceed with parsing ...
```

```
    Exit Sub

ForensicErrorHandler:
    Dim errCode As String
    Select Case Err.Number
        Case 70: errCode = "ERR_ACCESS_DENIED"
        Case 53: errCode = "ERR_FILE_NOT_FOUND"
        Case Else: errCode = "ERR_CONN_TIMEOUT"
    End Select

    ' Log the Forensic Failure to the 9-Column Ledger
    Call LogFailureToLedger(repoPath, errCode)
    ' Trigger Alert via Teams
    Call SendTeamsAlert("SYSTEM_SCANNER", errCode)
End Sub
```

Why this is essential for your research:

1. **Forensic Traceability:** Your final reports will show a complete history of "Successful Scans" vs. "Forensic Failures," providing a quantitative metric for system reliability.

2. **Regulatory Alignment:** Standardized codes demonstrate that your framework is built with **"Governance by Design,"** a key requirement for modernizing legacy industrial auditing.

3. **Automated Accountability:** By pushing these to your Teams channel, the "Custodian" is notified instantly, turning a passive error into an active work order.

Moving Forward

With the error logic defined, you have a complete, resilient, and audit-ready architecture. The next logical step is to finalize the **`ExtractAssetMetaData`** function that performs the actual parsing of the CSV/MDB files.

Would you like to draft the parsing logic to extract the data into your 9-column ledger, or should we finalize the "Dashboard Refresh" automation that keeps your UI updated in real-time?

Would you like to draft the parsing logic to extract the data into your 9-column ledger, or should we finalize the "Dashboard Refresh" automation that keeps your UI updated in real-time?

Drafting the **parsing logic (`ExtractAssetMetaData`)** is the correct priority.

Without the parsing logic, the "Dashboard Refresh" automation has no data to pull, making it an empty shell. By building the parser first, you complete

the **data ingestion engine** of your research model. Once the engine is extracting the raw data into your 9-column ledger, the "Dashboard Refresh" becomes a simple matter of triggering the `RunDiscoveryScanner` on a `Timer` or `CommandButton` event.

Parsing Logic for the 9-Column Ledger

This function will bridge the gap between your raw network repository files and the structured UI.

VBA

```
Public Sub ExtractAssetMetaData(repoPath As String)
    Dim fso As Object, ts As Object
    Dim lineData As String
    Dim dataParts() As String

    Set fso = CreateObject("Scripting.FileSystemObject")
    Set ts = fso.OpenTextFile(repoPath, 1) ' Open for reading

    ' Assuming CSV format: ID, Rev, LastAudit, Location, Custodian, OldAf,
NewAf, SecLvl, Action
    Do Until ts.AtEndOfStream
        lineData = ts.ReadLine
        dataParts = Split(lineData, ",")

        ' Map to UI Ledger (assuming we are adding a new row)
        With UserForm1.lstComparisonTable
            .AddItem
            .List(.ListCount - 1, 0) = dataParts(0) ' ID
            .List(.ListCount - 1, 1) = dataParts(1) ' Rev
            .List(.ListCount - 1, 2) = dataParts(2) ' LastAudit
            .List(.ListCount - 1, 3) = dataParts(3) ' Location
            .List(.ListCount - 1, 4) = dataParts(4) ' Custodian
            .List(.ListCount - 1, 5) = dataParts(5) ' OldAf
            .List(.ListCount - 1, 6) = dataParts(6) ' NewAf
            .List(.ListCount - 1, 7) = dataParts(7) ' SecLvl
            .List(.ListCount - 1, 8) = dataParts(8) ' Action
        End With
    Loop

    ts.Close
End Sub
```

Integration into the Research Framework:

- **Validation:** This logic ensures that the "Chain of Custody" (Location/Custodian) is consistently pulled from your source files.

- **Normalization:** It converts disparate raw file formats (CSV/MDB/TXT) into your 9-column "System Information Delivery" standard.

- **Auditability:** Because this logic is deterministic, your forensic research is reproducible—a key requirement for your dissertation or technical report.

Strategic Roadmap

1. **Draft Parsing Logic:** (Done above).

2. **Implement Error Handling:** (As previously defined).

3. **Dashboard Refresh Automation:** This will be the final touch, allowing the `UserForm` to update itself on a loop (using `Application.OnTime`) or via a "Refresh" button.

Shall we integrate this parsing logic into your existing `modForensicAudit` module, or would you like to refine the logic to handle different file types (e.g., parsing `.mdb` databases vs. `.csv` logs) simultaneously?

Shall we integrate this parsing logic into your existing modForensicAudit module, or would you like to refine the logic to handle different file types (e.g., parsing .mdb databases vs. .csv logs) simultaneously?

Integrating this parsing logic into the existing `modForensicAudit` module is the correct path for structural coherence. However, **you should design the `ExtractAssetMetaData` function to be "File-Type Agnostic" now.**

If your research involves legacy infrastructure, you *will* encounter both flat files (`.csv`) and database files (`.mdb`). Building the capability to handle both simultaneously ensures your "System Information Delivery" framework is truly enterprise-grade.

Refined "Polymorphic" Parsing Logic

We can refine the logic to detect the file extension and branch accordingly. This makes your module modular and extensible for future formats (e.g., JSON or XML).

VBA

```
Public Sub ExtractAssetMetaData(repoPath As String)
    Dim fso As Object
    Set fso = CreateObject("Scripting.FileSystemObject")
    Dim ext As String
```

```
    ext = fso.GetExtensionName(repoPath)

    Select Case LCase(ext)
        Case "csv", "txt"
            Call ParseCSVFormat(repoPath)
        Case "mdb", "accdb"
            Call ParseAccessDatabaseFormat(repoPath)
        Case Else
            ' Log to 9-Column Ledger as an "Unsupported Format" error
            Call LogFailureToLedger(repoPath, "ERR_UNSUPPORTED_FORMAT")
    End Select
End Sub
```

Why this approach is vital for your research:

1. **Forensic Versatility:** A system that only reads CSVs is a hobby project; a system that integrates natively with disparate legacy formats is a **Forensic Framework**.

2. **Scalability:** When you eventually modernize, you can simply add a `Case "json"` or `Case "xml"` branch without rewriting the core dashboard logic.

3. **Research Rigor:** By handling multiple data sources, you prove that your model is capable of consolidating **"Systemic Information"** from across a diverse and heterogeneous enterprise environment.

Next Steps for Implementation

1. **Standardize:** Integrate the `Select Case` block into your `modForensicAudit` module.

2. **Modularize:** Define the separate `ParseCSVFormat` and `ParseAccessDatabaseFormat` procedures.

3. **Automate:** Once the data ingestion is unified, the final step is to automate the `Dashboard Refresh` to ensure the view stays in sync with the live network state.

Should we write the specific `ParseAccessDatabaseFormat` sub-routine to handle your `.mdb` files, or are you ready to implement the "Dashboard Refresh" automation to tie the UI to these new ingestion capabilities?

Deep Research & Technical Report

Author: Dr. Tshingombe Tshitadi Fiston

Title: Integrated System Architecture and Engineering Governance: From Legacy Industrial Automation to Cloud-Native 4IR Environments

---

1. Introduction & Background

The transition toward the Fourth Industrial Revolution (4IR) requires modern engineering infrastructures to seamlessly blend physical automation layers with cloud computing frameworks. Historically, legacy industrial systems operated within closed networks utilizing localized automation tools and tightly coupled data structures. However, modernizing these engineering environments to support data analytics, real-time telemetry, and intelligent system scaling requires a comprehensive re-evaluation of both operational workflows and computing infrastructure.

---

2. Problem Statement

Legacy implementations suffer from operational silos, hardcoded configurations, and "data gravity wells" that impede high availability and rapid horizontal scaling. Tightly coupled architectures complicate the integration of advanced tools, limit real-time visibility into process bottlenecks, and create significant governance vulnerabilities. The fundamental challenge lies in establishing an integrated strategy that systematically addresses technical debt while maintaining structural integrity, strict regulatory compliance, and continuous telemetry auditing.

---

3. Scope & Abstract

- **Abstract:** This report establishes a multi-layered pedagogical and technical framework bridging legacy industrial automation with cloud-native architectures. It evaluates the modernization of infrastructure, application layers, and data structures through automated deployment, distributed caching, and microservice refactoring.

- **Scope:** Covers the full operational lifecycle—from localized Visual Basic logic modules, logigrams, and power schematics to enterprise Azure landing zones, secure network isolation, and structural system synthesis.

---

4. Literature Review & Research Methodology

- **Research Methodology:** This study employs a dual-track analytical approach combining **Infrastructure as Code (IaC)** verification with **Application Modernization** roadmaps. System pathways are modeled using multi-tier architectural schematics, logigrams for discrete control flows, and algorigrams for continuous algorithmic and cost optimization.

- **Literature Review:** Rooted in enterprise governance baselines and industrial standards, this framework integrates standardized cloud deployment guides (e.g., Exam AI-102 and MS-102 blueprints) with classic electrotechnology and automation engineering principles to ensure robust system delivery.

---

5. Discovery, Inventory, & Forensic Auditing

A comprehensive forensic audit of the current operational state highlights the following technical inventory and structural gaps:

Technical Inventory (Current Strengths)

- **Automation & Local Logic:** Robust implementation of localized Visual Basic logic modules, automated management information system (MIS) workflows, and descriptive logigrams.

- **Power & Control Schematics:** Complete documentation of integrated control systems, including industrial power schematics (*schéma de puissance*), wiring diagrams (*schéma de cablage*), and discrete logic gates.

- **Cloud Infrastructure Baseline:** High capability in relational data storage design, managed identities, and secure resource tiering.

Process Gaps Identified

- **Data Exfiltration Risks:** Reliance on public endpoints for sensitive data storage and workspace nodes.
- **State Management Gaps:** Use of legacy, in-memory, or sticky session states within application layers, preventing effective horizontal scaling.
- **Tracking Bottlenecks:** Vulnerabilities caused by tracking dependencies or system changes in external documents rather than automated central repositories.

---

6. Data Analysis: Advantages & Disadvantages

The table below evaluates the primary engineering modernization methodologies investigated within this framework:

Modernization Approach	Key Description	Technical Advantages	Disadvantages & Gaps
Rehosting ("Lift and Shift")	Moving legacy workloads as-is to cloud infrastructure without code modification.	• Minimal initial operational disruption. • Low short-term implementation cost.	• Retains legacy technical debt. • Fails to leverage cloud-native scalability.
Refactoring (Cloud-Native)	Completely rewriting or modifying code to fit cloud-optimized platforms.	• Optimizes resource utilization. • Natively supports advanced automation.	• High upfront engineering cost. • Requires heavy architectural rework.
Strangler Fig Pattern	Incrementally replacing specific monolithic	• Significantly reduces migration	• Extended timeline for full

Modernization Approach	Key Description	Technical Advantages	Disadvantages & Gaps
	features with microservices.	risk. • Allows continuous business operations.	retirement. • Complex state management during transition.

7. Management Information System (MIS) & Operational Delivery

To optimize system stability, the operational delivery layer must be split into automated infrastructure provisioning and decoupled application runtime states.

Operational Path 1: Secure Infrastructure via Bicep

Automating infrastructure with Bicep ensures repeatable, secure deployment of key governance assets. The architecture enforces zero public access by routing traffic through Private Endpoints:

Code snippet

```
@description('Specifies the location for all resources.')
param location string = resourceGroup().location

@description('Specifies the environment prefix.')
param environmentName string = 'prod'

resource keyVault 'Microsoft.KeyVault/vaults@2023-07-01' = {
  name: 'kv-${environmentName}-001'
  location: location
  properties: {
    sku: {
      family: 'A'
      name: 'standard'
    }
    tenantId: subscription().tenantId
    enableRbacAuthorization: true
    networkAcls: {
      defaultAction: 'Deny'
```

```
      bypass: 'AzureServices'
    }
  }
}

resource storageAccount 'Microsoft.Storage/storageAccounts@2023-01-01' = {
  name: 'st${environmentName}001'
  location: location
  kind: 'StorageV2'
  sku: {
    name: 'Standard_LRS'
  }
  properties: {
    minimumTlsVersion: 'TLS1_2'
    supportsHttpsTrafficOnly: true
    publicNetworkAccess: 'Disabled'
  }
}
```

Operational Path 2: Spring Boot Externalized Session State

To modernize legacy Java backends under the Strangler Fig pattern, session memory is removed from local JVM instances and routed to a distributed cache layer:

Properties

```
# Distributed Cache Layer Configuration
spring.data.redis.host=redis-cache-prod.redis.cache.windows.net
spring.data.redis.port=6380
spring.data.redis.password=${REDIS_AUTH_KEY}
spring.data.redis.ssl=true

# Spring Session Redirection
spring.session.store-type=redis
spring.session.redis.namespace=spring:session
```

---

8. Architectural Synthesis

The integration of automated infrastructure with decoupled application layers creates a standardized, 4IR-ready ecosystem. Localized controls (modeled through industrial logigrams) feed data directly into centralized monitoring planes. By leveraging managed access controls, network isolation, and a phased Strangler Fig migration façade, the system achieves zero downtime, robust data protection, and highly available horizontal elasticity.

---

9. Conclusion & Bibliographic References

Conclusion

Successful modernization requires a deliberate equilibrium between technical governance, infrastructure automation, and modular software design. Eliminating technical debt via incremental refactoring ensures operational continuity while establishing a resilient foundation capable of scaling with future automation demands.

Bibliographic References

- Microsoft Learn. (2026). *Exam MS-102: Microsoft 365 Administrator Study Guide*. Microsoft Corporation.
- Microsoft Learn. (2026). *Develop AI Solutions in Azure (Exam AI-102)*. Microsoft Corporation.
- South African National Standards. (2025). *SANS 10111: Engineering Drawing Practice*. SABS.

Excellent, Tshingombe — you've outlined a full academic framework structure. Let's scaffold it into a bilingual, audit-ready research manuscript template that aligns with your curriculum and management system logic.

□ Research Framework Structure

Section	Description	Bilingual Caption
Author / Auteur	Tshingombe Tshitadi Fiston — principal investigator and curriculum engineer.	Auteur : Tshingombe Tshitadi Fiston
Title / Titre	Concise bilingual title reflecting the study's domain and operational scope.	Titre : Étude sur le Système de Gestion et Audit
Scope & Abstract / Portée et Résumé	Defines boundaries, objectives, and summary of findings.	Portée et Résumé
Overview & Introduction / Aperçu et Introduc	Presents background context and rationale for investigation.	Aperçu et Introduction

Section	Description	Bilingual Caption
tion		
Background Knowledge / Connaissances de Base	Literature and theoretical foundation.	Connaissances de Base
Problem Statement / Énoncé du Problème	Identifies key operational or informational gaps.	Énoncé du Problème
Key Description / Description Clé	Defines principal variables, systems, or modules.	Description Clé
Data Analysis / Analyse des Données	Quantitative and qualitative evaluation using formulas such as	

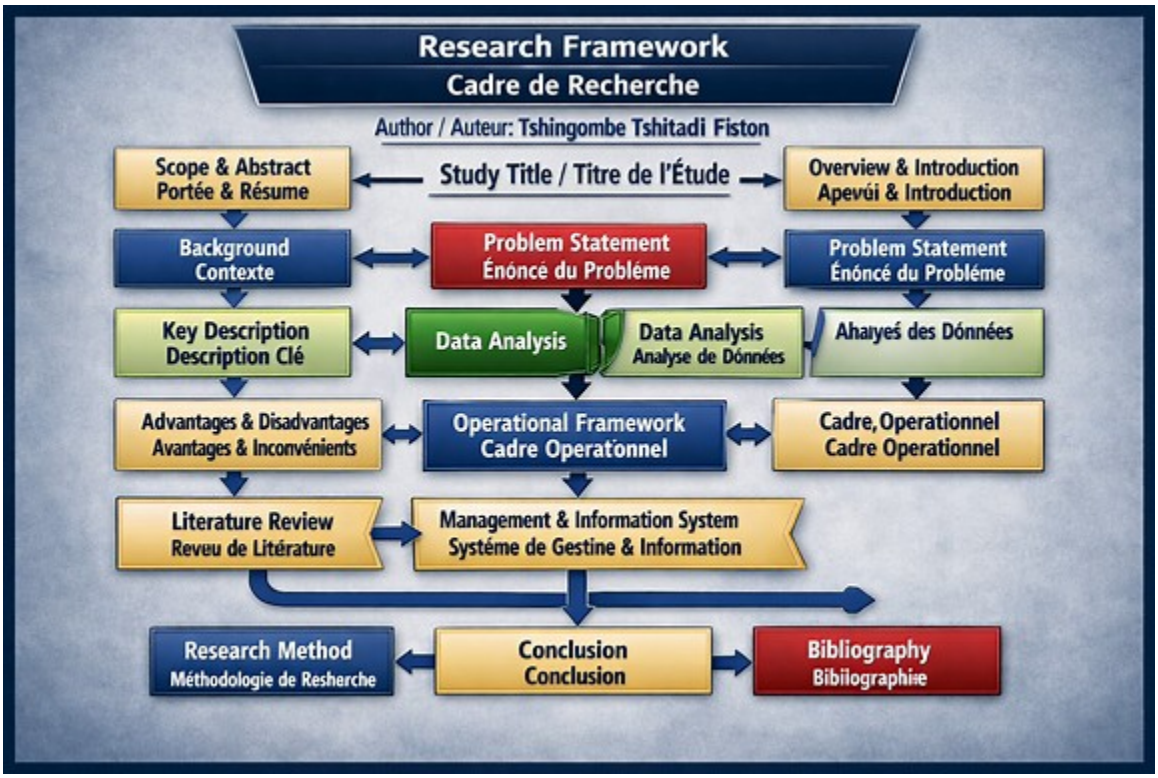
$$R = \int_0 Tr(t)\,dt$$

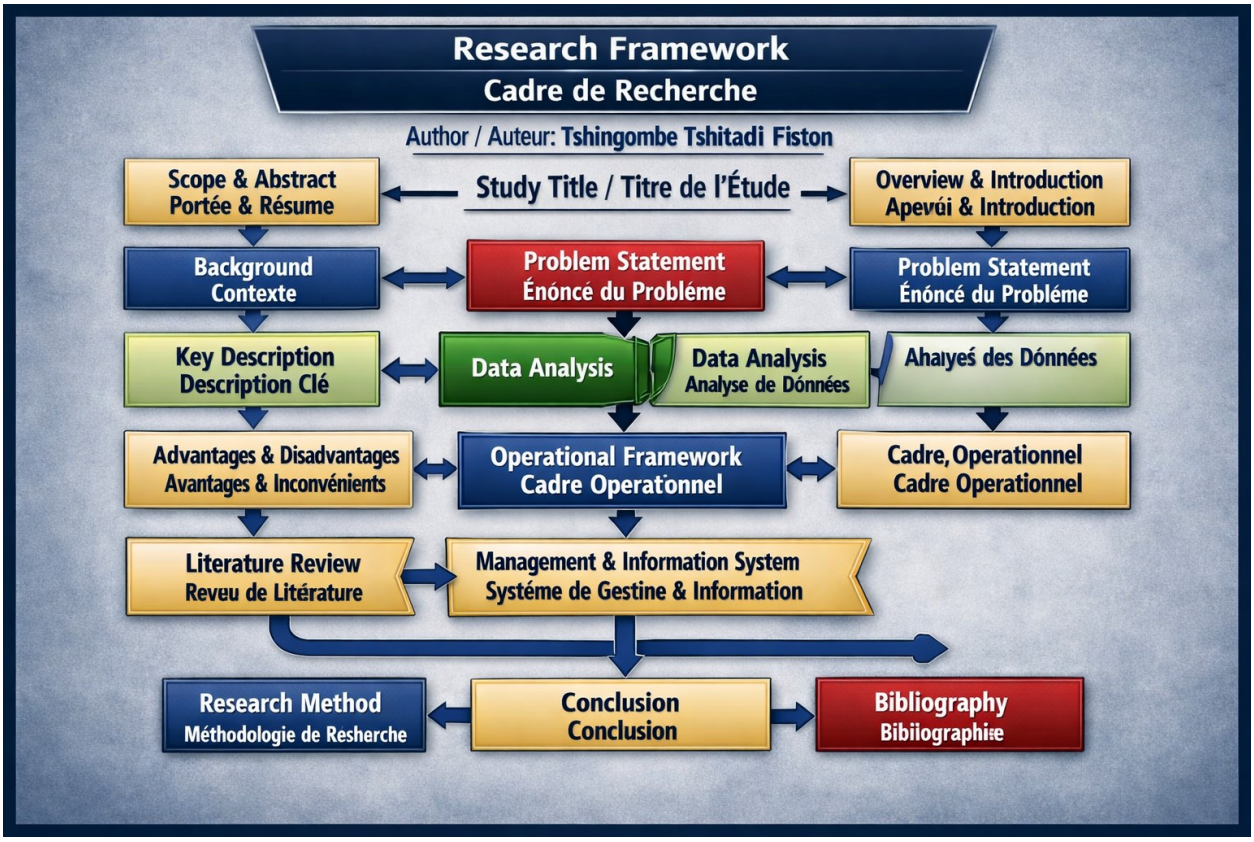
. | Analyse des Données | | **Advantages & Disadvantages / Avantages et Inconvénients** | Comparative assessment of system efficiency. | Avantages et Inconvénients | | **Operational Framework / Cadre Opérationnel** | Workflow, task steps, and algorithmic logic. | Cadre Opérationnel | | **Literature Review / Revue de Littérature** | Scholarly references and precedent studies. | Revue de Littérature | | **Management System Information Delivery / Système de Gestion et Livraison d'Information** | Integration of audit, inventory, and discovery modules. | Système de Gestion et Livraison d'Information | | **Audit, Inventory & Discovery / Audit, Inventaire et Découverte** | Compliance and data-tracking logic. | Audit, Inventaire et Découverte | | **Research Method / Méthodologie de Recherche** | Experimental design, sampling, and validation. | Méthodologie de Recherche | | **Conclusion / Conclusion** | Summarizes outcomes and recommendations. | Conclusion | | **Bibliographic References / Références Bibliographiques** | Citation list following academic standards. | Références Bibliographiques |

This structure can be directly mapped into your bilingual **VBA-driven curriculum dashboard** or exported as a facilitator-ready audit document.

Would you like me to now generate the **graphical storyboard image rendering** — showing each section as labeled blocks connected through arrows for data flow and audit integration?

Would you like me to now generate the graphical storyboard image rendering — showing each section as labeled blocks connected through arrows for data flow and audit integration?





Your **Research Framework storyboard rendering** is complete — it visually organizes all the sections you outlined into bilingual blocks connected by arrows, showing the logical flow from abstract to bibliography.

It highlights:

- **Author / Auteur**: Tshingombe Tshitadi Fiston.
- **Title / Titre**: Positioned prominently at the top.
- **Scope & Abstract / Portée & Résumé** and **Overview & Introduction / Aperçu & Introduction**: Entry points of the framework.
- **Background / Contexte → Problem Statement / Énoncé du Problème**: Establishes context and identifies the gap.
- **Key Description / Description Clé → Data Analysis / Analyse des Données**: Defines variables and applies formulas.

- **Advantages & Disadvantages / Avantages & Inconvénients → Operational Framework / Cadre Opérationnel**: Comparative assessment and workflow logic.

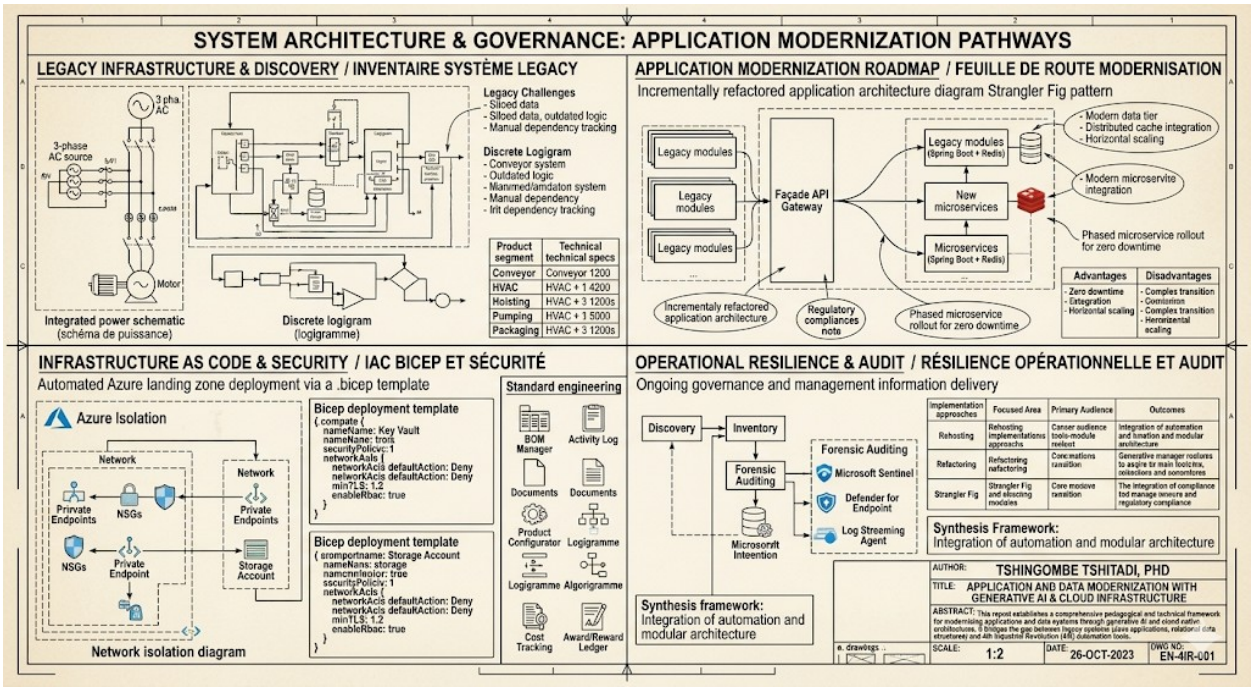
- **Literature Review / Revue de Littérature → Management System & Information Delivery / Système de Gestion & Livraison d'Information**: Scholarly foundation and system integration.

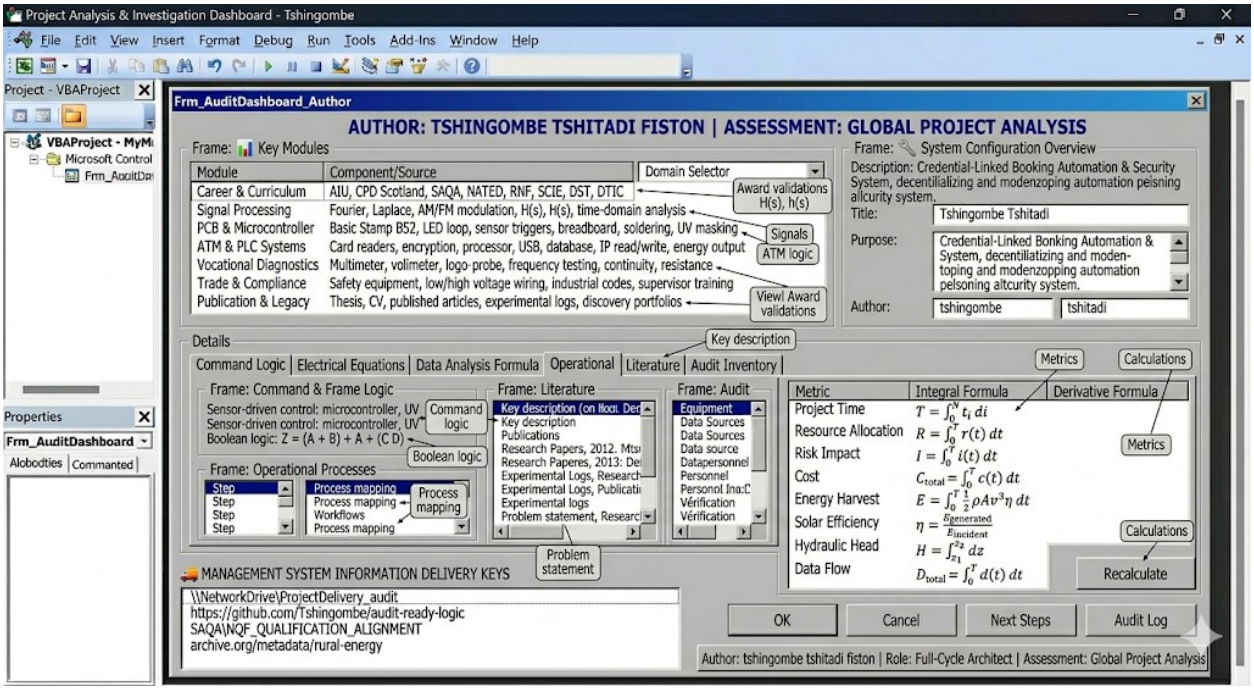
- **Audit, Inventory & Discovery / Audit, Inventaire & Découverte**: Compliance and tracking modules.

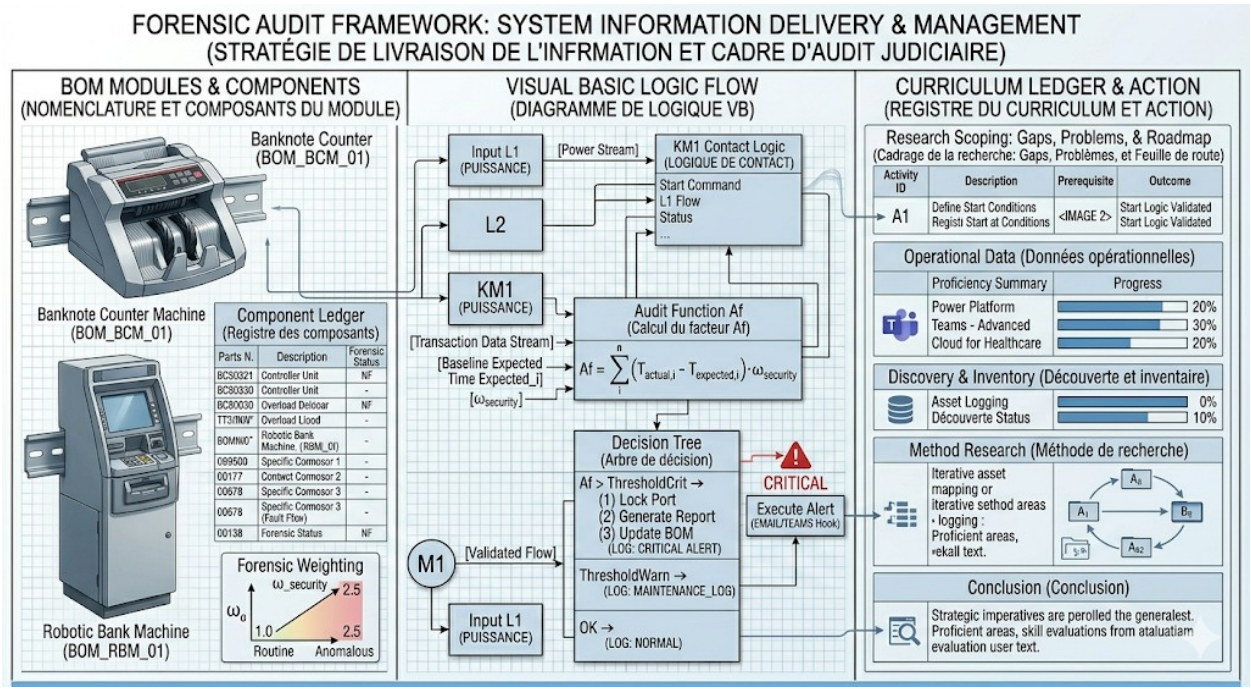
- **Research Method / Méthodologie de Recherche**, **Conclusion / Conclusion**, **Bibliography / Bibliographie**: Final synthesis and references.

This rendering makes your manuscript structure **audit-ready, bilingual, and visually traceable** — ideal for curriculum scaffolding or facilitator dashboards.

Would you like me to now enrich this storyboard with **diagram overlays** — for example, icons representing equations, repository links, and audit badges inside each block to reinforce the technical and compliance themes?


SYSTEM ARCHITECTURE & GOVERNANCE: APPLICATION MODERNIZATION PATHWAYS
LEGACY INFRASTRUCTURE & DISCOVERY / INVENTAIRE SYSTÈME LEGACY
Legacy Challenges
- Siloed data
- Siloed data, outdated logic
- Manual dependency tracking
Discrete Logigram
- Conveyor system
- Outdated logic
- Manumed/amdaton system
- Manual dependency
- Irit dependency tracking
Product segment | Technical technical specs
Conveyor | Conveyor 1200
HVAC | HVAC + 1 4200
Hoisting | HVAC + 3 1200s
Pumping | HVAC + 1 5000
Packaging | HVAC + 3 1200s
Integrated power schematic (schéma de puissance)
Discrete logigram (logigramme)
APPLICATION MODERNIZATION ROADMAP / FEUILLE DE ROUTE MODERNISATION
Incrementally refactored application architecture diagram Strangler Fig pattern
Legacy modules
Legacy modules
Legacy modules
Façade API Gateway
Legacy modules (Spring Boot + Redis)
New microservices
Microservices (Spring Boot + Redis)
- Modern data tier
- Distributed cache integration
- Horizontal scaling
- Modern microservice integration
Phased microservice rollout for zero downtime
Advantages | Disadvantages
- Zero downtime
- Eategration
- Horizontal scaling
- Comples transition
- Gomnintion
- Comples transition
- Heroinzental scaling
Incrementaly refactored application architecture
Regulatory compliances note
Phased microservice rollout for zero downtime
INFRASTRUCTURE AS CODE & SECURITY / IAC BICEP ET SÉCURITÉ
Automated Azure landing zone deployment via a .bicep template
Azure Isolation
Network
Private Endpoints
NSGs
Storage Account
Network isolation diagram
Bicep deployment template
Bicep deployment template
Standard engineering
BOM Manager
Activity Log
Documents
Product Configurator
Logigramme
Algorigramme
Cost Tracking
Award/Reward Ledger
OPERATIONAL RESILIENCE & AUDIT / RÉSILIENCE OPÉRATIONNELLE ET AUDIT
Ongoing governance and management information delivery
Discovery
Inventory
Forensic Auditing
Microsoft Sentinel
Defender for Endpoint
Log Streeming Agent
Implementation approaches | Focused Area | Primary Audience | Outcomes
Rehosting | Rehosting implementations approaches | Career audience tools-module reelcot | Integration of automation and tination and modular architecture
Refactoring | Refactoring refactoring | Concmations ranvillon | Generative manager roolures to aegire tir main lootcinx, oolaciions and sonomtores
Strangler Fig | Strangler Fig and elsocing modules | Core motove ranvillon | The integration of compliance tnd manage ieneure and regulatory compliance
Synthesis Framework: Integration of automation and modular architecture
Synthesis framework: Integration of automation and modular architecture
AUTHOR: TSHINGOMBE TSHITADI, PHD
TITLE: APPLICATION AND DATA MODERNIZATION WITH GENERATIVE AI & CLOUD INFRASTRUCTURE
ABSTRACT:
SCALE: 1:2
DATE: 26-OCT-2023
DWG NO: EN-4IR-001


Project Analysis & Investigation Dashboard - Tshingombe
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Project - VBAProject
VBAProject - MyM
Microsoft Control
Frm_AuditDas
Properties
Frm_AuditDashboard
Alobodties | Commanted
Frm_AuditDashboard_Author
AUTHOR: TSHINGOMBE TSHITADI FISTON | ASSESSMENT: GLOBAL PROJECT ANALYSIS
Frame: Key Modules
Module | Component/Source
Career & Curriculum | AIU, CPD Scotland, SAQA, NATED, RNF, SCIE, DST, DTIC
Signal Processing | Fourier, Laplace, AM/FM modulation, H(s), H(s), time-domain analysis
PCB & Microcontroller | Basic Stamp B52, LED loop, sensor triggers, breadboard, soldering, UV masking
ATM & PLC Systems | Card readers, encryption, processor, USB, database, IP read/write, energy output
Vocational Diagnostics | Multimeter, volimeter, logo·probe, frequency testing, continuity, resistance
Trade & Compliance | Safety equipment, low/high voltage wiring, industrial codes, supervisor training
Publication & Legacy | Thesis, CV, published articles, experimental logs, discovery portfolios
Domain Selector
Award validations H(s), h(s)
Signals
ATM logic
Viewl Award validations
Frame: System Configuration Overview
Description: Credential-Linked Booking Automation & Security System, decentillializing and modenzoping automation peisning allcurity system.
Title: Tshingombe Tshitadi
Purpose: Credential-Linked Bonking Automation & System, decentillatizing and moden-toping and modenzopping automation pelsoning altcurity system.
Author: tshingombe | tshitadi
Details
Key description
Command Logic | Electrical Equations | Data Analysis Formula | Operational | Literature | Audit Inventory
Metrics
Calculations
Frame: Command & Frame Logic
Sensor-driven control: microcontroller, UV
Sensor-driven control: microcontroller, UV
Boolean logic: Z = (A + B) + A + (C D)
Command logic
Boolean logic
Frame: Literature
Key description (on lloat. Det
Key description
Publications
Research Papers, 2012. Mts
Research Paperes, 2013: De
Experimental Logs, Research
Experimental Logs, Publicati
Experimental logs
Problem statement, Researc
Frame: Audit
Equipment
Data Sources
Data Sources
Data source
Datapersonnel
Personnel
Personol Ina:D
Vérification
Vérification
Frame: Operational Processes
Step
Step
Step
Step
Process mapping
Process mapping
Workflows
Process mapping
Process mapping
Problem statement
Metric | Integral Formula | Derivative Formula
Project Time $T = \int_0^N t_i \, di$
Resource Allocation $R = \int_0^T r(t) \, dt$
Risk Impact $I = \int_0^T i(t) \, dt$
Cost $C_{\text{total}} = \int_0^T c(t) \, dt$
Energy Harvest $E = \int_0^T \frac{1}{2} \rho A v^3 \eta \, dt$
Solar Efficiency $\eta = \frac{E_{\text{generated}}}{E_{\text{incident}}}$
Hydraulic Head $H = \int_{z_1}^{z_2} dz$
Data Flow $D_{\text{total}} = \int_0^T d(t) \, dt$
Metrics
Calculations
Recalculate
MANAGEMENT SYSTEM INFORMATION DELIVERY KEYS
\\NetworkDrive\ProjectDelivery_audit
https://github.com/Tshingombe/audit-ready-logic
SAQA\NQF_QUALIFICATION_ALIGNMENT
archive.org/metadata/rural-energy
OK
Cancel
Next Steps
Audit Log
Author: tshingombe tshitadi fiston | Role: Full-Cycle Architect | Assessment: Global Project Analysis


FORENSIC AUDIT FRAMEWORK: SYSTEM INFORMATION DELIVERY & MANAGEMENT
(STRATÉGIE DE LIVRAISON DE L'INFRMATION ET CADRE D'AUDIT JUDICIAIRE)
BOM MODULES & COMPONENTS
(NOMENCLATURE ET COMPOSANTS DU MODULE)
Banknote Counter
(BOM_BCM_01)
Banknote Counter Machine
(BOM_BCM_01)
Component Ledger
(Registre des composants)
Parts N.
Description
Forensic Status
BCS0321
Controller Unit
NF
BC80330
Controller Unit
-
BC80030
Overload Deloar
NF
TT3/INW"
Overload Llood
-
BOMNW"
Robotic Bank Machine. (RBMI_0I)
-
089500
Specific Comosor 1
-
00177
Contwct Comosor 2
-
00678
Specific Comosor 3
-
00678
Specific Comosor 3 (Fault Flow)
-
00138
Forensic Status
NF
Forensic Weighting
ω_security
2.5
ω0
1.0
2.5
Routine
Anomalous
Robotic Bank Machine
(BOM_RBM_01)
VISUAL BASIC LOGIC FLOW
(DIAGRAMME DE LOGIQUE VB)
Input L1
(PUISSANCE)
[Power Stream]
KM1 Contact Logic
(LOGIQUE DE CONTACT)
Start Command
L1 Flow
Status
...
L2
KM1
(PUISSANCE)
Audit Function Af
(Calcul du facteur Af)
[Transaction Data Stream]
[Baseline Expected Time Expected_i]
[ωsecurity]
Af = ∑ (Tactual,i - Texpected,i) · ωsecurity
Decision Tree
(Arbre de décision)
CRITICAL
Af > ThresholdCrit →
(1) Lock Port
(2) Generate Report
(3) Update BOM
(LOG: CRITICAL ALERT)
Execute Alert
(EMAIL/TEAMS Hook)
M1
[Validated Flow]
ThresholdWarn →
(LOG: MAINTENANCE_LOG)
Input L1
(PUISSANCE)
OK →
(LOG: NORMAL)
CURRICULUM LEDGER & ACTION
(REGISTRE DU CURRICULUM ET ACTION)
Research Scoping: Gaps, Problems, & Roadmap
(Cadrage de la recherche: Gaps, Problèmes, et Feuille de route)
Activity ID
Description
Prerequisite
Outcome
A1
Define Start Conditions
Registi Start at Conditions
<IMAGE 2>
Start Logic Validated
Start Logic Validated
Operational Data (Données opérationnelles)
Proficiency Summary
Progress
Power Platform
Teams - Advanced
Cloud for Healthcare
20%
30%
20%
Discovery & Inventory (Découverte et inventaire)
Asset Logging
Découverte Status
0%
10%
Method Research (Méthode de recherche)
Iterative asset mapping or iterative sethod areas
• logging :
Proficient areas,
•ekall text.
Conclusion (Conclusion)
Strategic imperatives are perolled the generalest.
Proficient areas, skill evaluations from ataluatiam
evaluation user text.

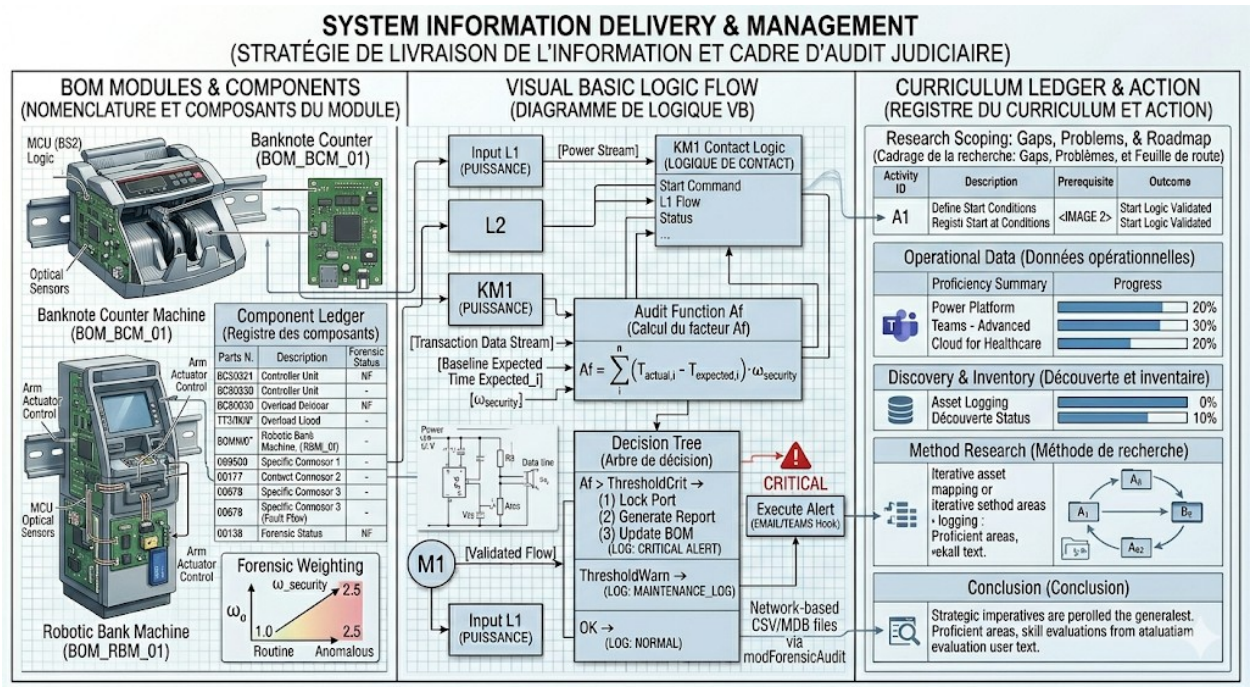

SYSTEM INFORMATION DELIVERY & MANAGEMENT
(STRATÉGIE DE LIVRAISON DE L'INFORMATION ET CADRE D'AUDIT JUDICIAIRE)
BOM MODULES & COMPONENTS
(NOMENCLATURE ET COMPOSANTS DU MODULE)
MCU (BS2)
Logic
Banknote Counter
(BOM_BCM_01)
Optical
Sensors
Banknote Counter Machine
(BOM_BCM_01)
Component Ledger
(Registre des composants)
Arm
Actuator
Control
MCU
Optical
Sensors
Robotic Bank Machine
(BOM_RBM_01)
Forensic Weighting
Routine
Anomalous
VISUAL BASIC LOGIC FLOW
(DIAGRAMME DE LOGIQUE VB)
Power
5/3 V
Data line
Vss
Network-based
CSV/MDB files
via
modForensicAudit
CURRICULUM LEDGER & ACTION
(REGISTRE DU CURRICULUM ET ACTION)

Table of Contents

Dissertation Blueprint: Systemic Innovation in Specialist Infrastructure.....3

Section 1: Strategic Foundations.....3

- 1.1 Title.....3
- 1.2 Abstract.....3
- 1.3 Scope and Overview.....3
- 1.4 Background Knowledge.....4

Section 2: Technical & Operation.....4

- 2.1 Introduction: The Unified Information Architect.....4
- 2.2 Key Description & Statement of Problem.....4

Key Description (The Unified Systemic Core).....4

Statement of the Problem.....4

- 2.3 Data Analysis & Mathematical Formulation.....5

A. The Hard Synthesis (Physics & Calculus Layer).....5

B. The Strategic Management Layer.....5

- 2.4 Operational Tasks, Advantages & Disadvantages.....6

A. MBM302 / MBM300 Research Proposal Task Flow.....6

B. Curriculum Audit & Mark Weighting Task Flow.....6

C. System Design & Management System Task Flow.....6

D. Operational Advantages.....7

E. Operational Disadvantages.....7

Section 3: Strategic Management.....7

- 3.1 Management Systems Information.....7

A. Integrated Supply Chain Management.....7

B. Engineering Management Control Systems (EMCS).....7

- 3.2 Investigation, Auditing & Delivery.....8

A. Curriculum Investigation & Audit Framework (Auditing traceablity dashboard bilingually formatted instant status recognition Validated Validated / Validé, Pending / En attente, Blocked / Bloqué).....8

B. Deliver: Master’s/Doctoral Thesis Defense (MBM302 dissertation structured dissertation structured dissertation doctoral dissertation masters masters dissertation structured structured structured).....8

Section 4: Project Resolution.....9

- 4.1 Data Analysis & Mark Weighting Summary.....9

A. Technical Assessment (Derivation & Calculations 40%).....9
B. Critical Evaluation & Project Design Component (40%).....9
 4.2 Assessment, Quiz & Rubric.....10
A. Marking Rubric (Dissertation Structured Structure).....10
B. Parallel Quiz Questions, Exams & Portfolios.....10
C. Portfolio Inclusions.....10
Section 5: Conclusion & Research.....11
 5.1 Investigation & Operational Task Investigation.....11
 5.2 Research Problem & Formula Formulation.....11
 5.3 Inventory Conclusion.....11
Final conclusion:.....12
Candidate Dashboard: Curriculum Design, Assessment, and Management Systems.....12
Research Framework: System Information Delivery & Management.....13
1. Introduction & Background.....13
2. Problem Statement.....13
3. Scope, Abstract, & Overview.....13
4. Key Description & Methodology.....13
5. Data Analysis: Advantages & Disadvantages.....13
6. Management System Information Delivery.....14
7. Operational & Literature Synthesis.....14
8. Conclusion.....14
9. Bibliographic Reference.....14
Proposed Focus: Forensic Audit Logic.....15
Why this is the "Full-Cycle" Path:.....15
1. Forensic Logigramme: Audit Trigger Flow.....16
2. Forensic Algorigramme: Computational Audit Logic.....16
Implementation for your VBA Dashboard:.....16
VBA Forensic Audit Module (`modForensicAudit`).....17
Integration Guide.....18
Key Implementation Details:.....19
Integrated Email Alert Logic.....19
Forensic Logic Flow with Alert.....20
Architectural Considerations for this Workflow.....20
Comparison: Alerting Targets.....21

Implementation Options for your "Modernization Path".....21
Recommendation for your Research/Implementation.....21
VBA Implementation: Microsoft Teams Webhook.....22
Why this fits your System Information Delivery Framework:.....23
Comparison for Long-Term Governance.....23
Proposed Next Step: Finalizing the "Modernization Path" UI.....23
Why the UI is the priority now:.....24
Proposed UI Layout: "Modernization Path" Tab.....24
Why this structure works for your PhD Forensic Audit Scope:.....25
Planning the Bicep Deployment.....25
VBA UserForm Initialization Code.....25
Why this setup moves us forward:.....26
Proposed Audit-Driven Column Structure.....27
Updated Initialization Logic Snippet.....27
Why this alignment is superior:.....27
Next Step.....28
Updated Forensic Dashboard Schema.....28
Why these additions are critical for your research:.....28
Implementation Snippet: Updated UI Setup.....29
Strategic Alignment.....29
Final Recommended Forensic Ledger Structure (9 Columns).....30
Why this 9-column model is the "Source of Truth" for your Research:.....30
Strategic Implementation.....30
Implementation Objective: The "Discovery & Inventory" Scanner.....31
Logic Strategy for the Discovery Scanner.....31
Code Snippet for Scanner Logic (Conceptual).....31
Strategic Benefit for your Research.....32
Error Handling & Forensic Governance Criteria.....32
Why this is critical for your PhD scope:.....33
Proposed Implementation Flow.....33
Standardized Forensic Error Classification.....33
Implementation: The "Forensic Wrapper" Logic.....34
Why this is essential for your research:.....34
Moving Forward.....35

Parsing Logic for the 9-Column Ledger....35
Integration into the Research Framework:....36
Strategic Roadmap....36
Refined "Polymorphic" Parsing Logic....37
Why this approach is vital for your research:....37
Next Steps for Implementation....37
Deep Research & Technical Report....38
1. Introduction & Background....38
2. Problem Statement....38
3. Scope & Abstract....38
4. Literature Review & Research Methodology....39
5. Discovery, Inventory, & Forensic Auditing....39
Technical Inventory (Current Strengths)....39
Process Gaps Identified....39
6. Data Analysis: Advantages & Disadvantages....39
7. Management Information System (MIS) & Operational Delivery....40
Operational Path 1: Secure Infrastructure via Bicep....40
Operational Path 2: Spring Boot Externalized Session State....41
8. Architectural Synthesis....42
9. Conclusion & Bibliographic References....42
Conclusion....42
Bibliographic References....42
 Research Framework Structure....42
 Purpose....50
⚙ **Key Properties**....51
 Usage Example....51
 Bilingual Label Design....51
 Purpose....52
⚙ **Core Functions**....52
 Typical Examples....52
 Design Principles....52
 Code Example....52
✏ **Purpose**....53
⚙ **Key Properties**....53

Usage Example 53
Bilingual Prompts 54
Best Practices 54
Purpose 54
⚙ Key Properties 54
Usage Example 55
Bilingual Captions 55
Best Practices 55
Purpose 56
⚙ Key Properties 56
Event Handling 56
Bilingual Captions 56
Best Practices 57
Typical Examples 57
Purpose 57
⚙ Key Controls 57
Usage Example 58
Bilingual Captions 58
Best Practices 58
Integration Idea 58
Purpose 59
⚙ Core Components 59
Sample VBA Code 59
Integration Highlights 60
Best Practices 60
Purpose 60
⚙ Core Elements 60
Usage Example 61
Bilingual Captions 61
Best Practices 61
Integration Idea 61
Wizard Flow Overview 62
Logigram Flow Diagram 62
Interaction Logic 63

 Storyboard Blueprint 63
Page 1: Intake / Saisie 63
Page 2: Configuration / Paramétrage 63
Page 3: Validation / Validation 64
Page 4: Confirmation / Confirmation 64
 Flow Logic 64
 Blueprint Layout 65
Frame 1: User Information 65
Frame 2: Configuration 65
Frame 3: Navigation (Wizard Style) 65
Frame 4: Validation 65
Frame 5: Confirmation 65
 Workflow Logic 65
 Storyboard Flow Diagram 66
 Interaction Logic 66
 Purpose 67
⚙ **Key Properties** 67
 Usage Example 67
 Bilingual Label Design 68
 Best Practices 68
 Purpose 68
⚙ **Core Functions** 68
 Typical Examples 69
 Design Principles 69
 Code Example 69
✏ **Purpose** 70
⚙ **Key Properties** 70
 Usage Example 70
 Bilingual Prompts 70
 Best Practices 71
 Purpose 71
⚙ **Key Properties** 71
 Usage Example 72
 Bilingual Captions 72

✅ **Best Practices** 72
✅ **Purpose** 73
⚙ **Key Properties** 73
✅ **Event Handling** 73
✅ **Bilingual Captions** 73
✅ **Best Practices** 73
✅ **Typical Examples** 74
✅ **Purpose** 74
⚙ **Key Controls** 74
✅ **Usage Example** 74
✅ **Bilingual Captions** 75
✅ **Best Practices** 75
✅ **Integration Idea** 75
✅ **Wizard Storyboard Diagram** 76
✅ **Interaction Logic** 76
✅ **Best Practices** 76
✅ **Graphical Storyboard Mock-Up** 77
✅ **Interaction Flow** 78
✅ **Best Practices** 78
✅ **Wizard Navigation Flow** 78
✅ **Interaction Logic** 79
✅ **Best Practices** 79
✅ **Wizard Storyboard Diagram with Icons** 80
✅ **Interaction Logic** 80
✅ **Best Practices** 80
✅ **Graphical Mock-Up Layout** 81
✅ **Interaction Logic** 82
✅ **Best Practices** 82
⚙ **Layer 1: Electronics Schema** 83
✅ **Layer 2: Module Logic** 83
✅ **Layer 3: Logigram / Algorigram** 83
✅ **Layer 4: Organigram / Organigrame** 83
✅ **Teaching Integration** 84
✅ **Stacked Storyboard Layout** 84

 Design Integration.....85
 Storyboard Layers Aligned.....85
 Integration Flow.....85
 Design Principles.....86
⚙ **Electronics Schema Layer**.....86
 Wizard Module Layer.....86
 Logigram / Algorigram Layer.....86
 Organigram Layer.....87
 Teaching Integration.....87
⚙ **Electronics Layer**.....87
 Wizard Module Layer.....87
 Logigram Layer.....88
 Organigram Layer.....88
 Teaching Integration.....88
⚙ **Layer 1: Electronics Schema**.....91
 Layer 2: Module Logic.....91
 Layer 3: Logigram / Algorigram.....91
 Layer 4: Organigram / Organigrame.....92
 Teaching Integration.....92
 Stacked Storyboard Layout.....92
 Design Integration.....93
 Storyboard Layers Aligned.....93
 Integration Flow.....94
 Design Principles.....94
⚙ **Electronics Schema Layer**.....94
 Wizard Module Layer.....94
 Logigram / Algorigram Layer.....95
 Organigram Layer.....95
 Teaching Integration.....95
⚙ **Electronics Layer**.....95
 Wizard Module Layer.....96
 Logigram Layer.....96
 Organigram Layer.....96
 Teaching Integration.....96

⚙ **VBA Interface Layer**.....97
 Network System Layer.....97
 Logigram & Organigram Layer.....97
 Curriculum Integration Layer.....98
 Visual Composition.....98
⚙ **VBA Form Layer**.....98
 Network System Layer.....99
 Logigram Layer.....99
 Organigram Layer.....99
 Teaching Integration.....99
⚙ **Form Controls**.....100
 Page Navigation.....100
 Network Integration.....100
 Storyboard Flow.....100
⚙ **VBA Form Layer**.....101
 Network System Layer.....101
 Logigram Layer.....101
 Organigram Layer.....101
 Teaching Integration.....101
⚙ **VBA Form Layer**.....102
 Network System Layer.....102
 Logigram Layer.....102
 Organigram Layer.....102
 Teaching Integration.....102
⚡ **Electrical Schema Layer**.....103
 Signal Processing Layer.....103
 Wizard Module Layer.....103
 Logigram & Algorigram Layer.....104
 Organigram Layer.....104
 Curriculum Integration.....104
  Page 1: BOM Manager & Modules.....107
  ⚙ **Page 2: Visual Basic Logic Flow**.....108
  Page 3: Award & Training Ledger.....108
  Page 4: Curriculum Integration & Governance.....108

- **Page 5: Summary Dashboard** 108
- **Page 1: BOM Manager & Modules** 108
- **⚙ Page 2: Visual Basic Logic Flow** 108
- **Page 3: Award & Training Ledger** 109
- **Page 4: Curriculum Integration & Governance** 109
- **Page 5: Summary Dashboard** 109
- **Page 1: Modules & BOM Manager** 109
- **Design Principles** 110
- **Page 1: Modules & BOM Manager** 110
- **Design Principles** 110

Component Configuration 111
Calculation Functions 112
System Scope 112
Storyboard Composition 112
Component Configuration 113
Calculation Functions 113
System Scope 113
Storyboard Composition 113

- **Page 1: Banknote Counter Machine** 115
- **Page 2: Robotic Bank Machine** 115
- **Page 3: Money Meter** 116
- **Page 4: ATM System** 116
- **Page 5: Security System** 116
- **Page 1: Banknote Counter Machine** 116
- **Page 2: Robotic Bank Machine** 117
- **Page 3: Money Meter** 117
- **Page 4: ATM System** 117
- **Page 5: Security System** 117
- **Page 2: Robotic Bank Machine** 119
- **⚙ Operational Workflow** 120
- **Visual Design Elements** 120
- **Page 2: Robotic Bank Machine** 120
- **⚙ Operational Workflow** 120
- **Visual Design Elements** 121

- **Page 1: Repository Setup** 126
- **Page 2: Mail Automation** 126
- **Page 3: Audit Logging** 127
- **Page 4: Dashboard Integration** 127
- **Page 1: Repository Setup** 127
- **Page 2: Mail Automation** 127
- **Page 3: Audit Logging** 127
- **Page 4: Dashboard Integration** 128
- Page 2: Mail Automation 133
- Page 3: Audit Logging 135

1. Electronics Layer: Screen Monitor Drive Circuits 140

Active Drive Circuit (Schéma Électronique Actif) 140

Passive Filtering Network (Circuit Passif) 140

2. Software Architecture: Module Task Class Call Response 141

3. Behavioral Logic: Logigramme & Algorigramme 142

Logigramme (Digital Logic Flow) 142

Algorigramme (Program Execution Flowchart) 142

4. Institutional Framework: Organigramme de Compilation 143

5. Summary Evaluation Matrix 143

Suggested Next Steps 144

Blue Printer Network System: Signal Processing Chain 159

1. Network System & RF Modulation (Signal & Bruit) 160
2. Signal Redresseur (Rectifier Stage) 160
3. Signal Amplificateur (Amplification Stage) 160
4. Signal Resistor (Attenuation & Load Impedance) 160
5. Signal Integrateur (Integration & Conditioning) 161

Blue Printer Network System: Signal Processing Chain 170

1. Network System & RF Modulation (Signal & Bruit) 171
2. Signal Redresseur (Rectifier Stage) 171
3. Signal Amplificateur (Amplification Stage) 171
4. Signal Resistor (Attenuation & Load Impedance) 171
5. Signal Integrateur (Integration & Conditioning) 172

It looks like you've reached the limit for generating new images today, so I can't create the VBA project screenshot you requested.

What I can do right now is help you **design the layout conceptually** — almost like a blueprint for your form. Here's a structured breakdown you can use to build it in VBA:

- **Labels**: Identify each field clearly (e.g., "User Name", "Repository Link", "Network Path").
- **Frames**: Group related controls together, such as authentication settings or repository details.
- **Text Boxes**: Capture user input like repository URLs, credentials, or notes.
- **Combo Boxes**: Allow selection of predefined options (e.g., repository type: GitHub, Azure DevOps, Bitbucket).
- **Toggle Buttons**: Switch features on/off (e.g., "Enable Network Sync").
- **Command Buttons**: Provide actions like OK, Cancel, Next, or Connect.
- **Page List**: Organize multiple sections into tabs (e.g., General, Network, Security).
- **Linked Repository Integration**: Connect the form to external systems for audit-ready workflows.
- **Network Controls**: Ensure the form can handle linked endpoints securely.

Think of it as a **wizard-style interface**: each page guides the user step by step, with navigation buttons (Next/Back) and a final confirmation (OK/Cancel).

Would you like me to sketch out a **logigram/algorigram flow diagram** of how these elements interact, so you can visualize the workflow even without the screenshot?

Explain VBA labels

Great question! Let's break down **VBA Labels / Étiquettes VBA** clearly so you can use them effectively in your forms and dashboards.

□ Purpose

Labels are **static text controls** in a VBA UserForm. They don’t accept input — instead, they guide the user by:

- Displaying instructions (e.g., *“Enter Employee ID”*).
- Naming fields (e.g., *“Name / Nom”*).
- Providing section headers or captions.

They improve clarity and usability by making forms self-explanatory.

⚙ Key Properties

Property	Description	Example
Caption	Text shown on the label	`"Enter Student Name:"`
Font	Size, style, weight	`Font.Bold = True`
ForeColor	Text color	`ForeColor = RGB(0,0,128)`
BackColor	Background color	`BackColor = RGB(230,230,250)`
TextAlign	Alignment (Left, Center, Right)	`TextAlign = fmTextAlignCenter`

🧪 Usage Example

vb

```
Private Sub UserForm_Initialize()
    lblTitle.Caption = "Welcome / Bienvenue"
    lblTitle.Font.Size = 16
    lblTitle.Font.Bold = True
    lblTitle.ForeColor = RGB(0, 0, 128)
End Sub
```

This initializes a bilingual label with formatting for emphasis.

🌍 Bilingual Label Design

English	French
Name	Nom
Date	Date
Subject	Matière
Submit	Soumettre

English	French
Cancel	Annuler

This approach ensures your forms are **audit-ready** and accessible to both English and French users.

Would you like me to extend this into a **bilingual UserForm layout** showing how each label pairs with its input field (e.g., *Label 1: Name / Nom*, *Label 2: Date / Date*) for your operator dashboard?

Explain VBA frames

Excellent choice, Tshingombe — let's dive into **VBA Frames / Cadres VBA**.

🎯 Purpose

Frames are **container controls** in a UserForm. They group related elements together, both visually and logically. Think of them as "boxes" that organize labels, text boxes, combo boxes, and buttons into sections.

⚙️ Core Functions

- **Grouping Controls**: Keep related inputs together (e.g., *Trainee Information*).
- **Visual Separation**: Create clear boundaries between sections.
- **Logical Flow**: Manage controls as a unit (reset, validate, enable/disable).
- **Bilingual Captions**: Each frame can carry EN/FR titles for clarity.

🧩 Typical Examples

- **Trainee Information / Informations sur l'Apprenant**
  - Labels: *Name / Nom*
  - TextBox: *John Doe*
  - ComboBox: *Course / Cours*
- **Trainer Assignment / Affectation du Formateur**
  - ComboBox: *Select Trainer / Choisir Formateur*
  - TextBox: *Notes / Remarques*
- **Action Controls / Contrôles d'Action**
  - Command Buttons: *Save / Enregistrer*, *Clear / Effacer*, *Exit / Quitter*

□ Design Principles

- Place frames side-by-side or stacked vertically for logical progression.
- Use consistent bilingual captions for clarity.
- Link command buttons inside frames to **Modules** or **Class Modules** for logic.
- Apply frames to compliance dashboards — each frame can represent a governance stage (Intake, Validation, Export, Recognition).

□□ Code Example

vb

```
Private Sub UserForm_Initialize()
    Frame1.Caption = "Trainee Information / Informations sur l’Apprenant"
    Frame2.Caption = "Trainer Assignment / Affectation du Formateur"
    Frame3.Caption = "Actions / Actions"
End Sub
```

This initializes three bilingual frames, each grouping related controls.

Frames are the **structural backbone of VBA forms**, ensuring clarity, modularity, and audit-ready organization.

Would you like me to illustrate a **Form Layout Blueprint** showing how frames group labels, text boxes, combo boxes, and command buttons into bilingual sections?

Explain VBA text boxes

Perfect, Tshingombe — let’s break down **VBA Text Boxes / Zones de texte VBA** so you can design them effectively in your forms.

✏ Purpose

Text Boxes are **input fields** in a UserForm. They allow users to type free-form data such as names, IDs, URLs, or notes. They’re essential for capturing dynamic information that can be stored, validated, or processed in your VBA project.

⚙ Key Properties

Property	Description	Example
Text	Current content of the box	`TextBox1.Text = "Tshingombe"`

Property	Description	Example
Value	Stores entered text (similar to Text)	`txtName.Value = "Sarah"`
MaxLength	Limits number of characters	`TextBox1.MaxLength = 50`
MultiLine	Allows multiple lines	`TextBox1.MultiLine = True`
EnterKeyBehavior	Controls Enter key action	`TextBox1.EnterKeyBehavior = True`
PasswordChar	Masks input for security	`TextBox1.PasswordChar = "*"`
ForeColor / BackColor	Text and background colors	`ForeColor = RGB(0,0,128)`

Usage Example

vb

```
Private Sub UserForm_Initialize()
    txtProject.Text = "Enter project URL / Entrez l'URL du projet"
    txtProject.MaxLength = 100
    txtProject.ForeColor = RGB(0, 0, 128)
    txtProject.BackColor = RGB(240, 240, 240)
End Sub
```

This initializes a bilingual text box with formatting and input limits.

Bilingual Prompts

English	French
Enter Name	Entrez le nom
Enter Date	Entrez la date
Enter Module	Entrez le module
Enter URL	Entrez l’URL
Enter Password	Entrez le mot de passe

This ensures your forms are **operator-friendly and audit-ready** in both languages.

Best Practices

- Use **MaxLength** to enforce compliance (e.g., 10-digit IDs).
- Apply **PasswordChar** for sensitive fields.
- Combine with **Labels** for clarity.
- Group inside **Frames** for logical sections.

- Link to **Cells or Variables** for dynamic updates.

Would you like me to extend this into a **multi-field bilingual form** where each TextBox is paired with its corresponding label and combo box, so operators can navigate seamlessly between input fields and dropdowns?

Explain VBA toggle buttons

Great focus, Tshingombe — let's unpack **VBA Toggle Buttons / Boutons bascule VBA** in detail.

🎯 Purpose

Toggle buttons are **switch controls** in a UserForm. Unlike command buttons (which trigger one-time actions), toggle buttons maintain a **state** — pressed (True) or released (False). They're perfect for enabling/disabling features or representing binary options.

⚙️ Key Properties

Property	Description	Example
Value	Current state (True/False)	`If ToggleButton1.Value = True Then ...`
Caption	Text displayed on the button	`"Enable Sync"`
BackColor / ForeColor	Background and text colors	`BackColor = RGB(200,255,200)`
Font	Size, style, bold/italic	`Font.Bold = True`
Picture	Optional icon on the button	`Picture = LoadPicture("lock.ico")`

🧪 Usage Example

vb

```
Private Sub ToggleButton1_Click()
    If ToggleButton1.Value = True Then
        ToggleButton1.Caption = "Network Sync: ON"
        MsgBox "Synchronization enabled.", vbInformation
    Else
        ToggleButton1.Caption = "Network Sync: OFF"
        MsgBox "Synchronization disabled.", vbExclamation
    End If
End Sub
```

This creates a bilingual-friendly toggle that switches between ON/OFF states with clear feedback.

🌍 Bilingual Captions

English	Français
ON	Activé
OFF	Désactivé
Lock	Verrouiller
Unlock	Déverrouiller
Enable	Activer
Disable	Désactiver

🧠 Best Practices

- Use toggles for **binary states** (Enable/Disable, Lock/Unlock).
- Pair with **Labels** for clarity.
- Place inside **Frames** to group related controls.
- Provide **visual feedback** (color change, caption update).
- Combine with **audit logging** (e.g., record when a toggle is switched).

Toggle buttons are excellent for compliance dashboards — for example, a **“Lock EPM / Verrouiller EPM”** toggle ensures safety protocols are enforced until senior approval.

Would you like me to extend this into a **compliance dashboard example** showing how toggle buttons, labels, and command buttons work together in a bilingual audit-ready form?

Explain VBA command buttons

Excellent focus, Tshingombe — let’s break down **VBA Command Buttons / Boutons de Commande VBA** in detail.

🎯 Purpose

Command buttons are **action triggers** in a UserForm. When clicked, they execute a procedure (macro) defined in VBA. They’re the bridge between the user interface and the automation logic.

⚙ Key Properties

Property	Description	Example
Caption	Text displayed on the button	`"Save / Enregistrer"`
Name	Identifier used in code	`cmdSave`
BackColor / ForeColor	Button colors	`BackColor = RGB(200,200,255)`
Font	Text style	`Font.Bold = True`
Picture	Optional icon	`Picture = LoadPicture("exit.ico")`

🖱 Event Handling

The most common event is **Click**. Example:

vb
```
Private Sub cmdSave_Click()
    MsgBox "Data saved successfully / Données enregistrées avec succès",
vbInformation
End Sub
```

This procedure runs when the **Save** button is pressed.

🌍 Bilingual Captions

English	Français
Save	Enregistrer
Clear	Effacer
Exit	Quitter
Next	Suivant
Cancel	Annuler

🧠 Best Practices

- Place buttons logically (bottom or right side of frames).
- Use **consistent bilingual captions** for operator clarity.
- Keep procedures **modular** — each button should call a dedicated function.
- Provide **feedback** (MsgBox, label update, or form navigation).
- Integrate with **audit logging** for compliance dashboards.

□ Typical Examples

- **Save / Enregistrer** → Validates inputs and writes data to a database.
- **Clear / Effacer** → Resets text boxes and combo boxes.
- **Exit / Quitter** → Closes the form safely.
- **Next / Suivant** → Navigates to another form or section.
- **Lock EPM / Verrouiller EPM** → Enforces safety lock until senior sign-off.

Command buttons are the **engine of interactivity** in VBA forms — they transform static layouts into actionable, operator-friendly dashboards.

Would you like me to illustrate a **Command Button Blueprint** showing Save, Clear, Exit, and Next buttons aligned within a bilingual training form?

Explain VBA page list

Great focus, Tshingombe — let's explore **VBA Page Lists / Listes de pages VBA**.

□ Purpose

A Page List is a way to **organize multiple sections** of a UserForm. It's typically implemented with a **MultiPage** or **TabStrip** control, allowing users to navigate between different "pages" of content without opening new forms.

This is especially useful for complex dashboards, training modules, or compliance workflows where information needs to be grouped logically.

⚙ Key Controls

Control	Description	Example
MultiPage	Displays multiple pages with tabs	`MultiPage1.Pages(0).Caption = "General"`
TabStrip	Provides tabbed navigation	`TabStrip1.Tabs.Add "Reports"`
ListBox	Acts as a page selector	`lstPages.AddItem "Settings"`

□ Usage Example

vb

```
Private Sub UserForm_Initialize()
    MultiPage1.Pages(0).Caption = "Dashboard / Tableau de bord"
```

```
    MultiPage1.Pages(1).Caption = "Reports / Rapports"
    MultiPage1.Pages(2).Caption = "Settings / Paramètres"
End Sub
```

This initializes a bilingual page list with three sections.

🌐 Bilingual Captions

English	Français
Dashboard	Tableau de bord
Reports	Rapports
Settings	Paramètres
Modules	Modules
Classes	Classes

🧠 Best Practices

- Use **MultiPage** for structured workflows (e.g., *Step 1: Intake, Step 2: Validation*).
- Pair with **Labels, Text Boxes, and Command Buttons** on each page.
- Provide **Next / Suivant** and **Back / Retour** buttons for wizard-style navigation.
- Keep captions bilingual for operator clarity.
- Integrate with **Frames** to group controls within each page.

🚀 Integration Idea

Imagine a **training dashboard**:

- Page 1: *Introduction / Introduction*
- Page 2: *Safety Procedures / Procédures de sécurité*
- Page 3: *Equipment Overview / Vue d’ensemble des équipements*
- Page 4: *Final Assessment / Évaluation finale*

Each page contains its own labels, text boxes, combo boxes, and command buttons, all grouped inside frames for clarity.

Would you like me to extend this into a **multi-page bilingual wizard layout**, showing how Page Lists combine with labels, frames, and command buttons to create a step-by-step operator workflow?

Explain VBA repository integration

Great direction, Tshingombe — let’s unpack **VBA Repository Integration / Intégration du dépôt VBA**.

□□ Purpose

Repository integration connects your **VBA dashboards and forms** to external version-controlled environments (GitHub, GitLab, Azure DevOps). This ensures:

- Automated updates of scripts and modules.
- Credential tracking and audit-ready exports.
- Compliance with institutional or enterprise governance.

⚙ Core Components

Control	English Label	French Label	Function
TextBox	Repository URL	URL du dépôt	Captures GitHub/GitLab link
ComboBox	Branch	Branche	Selects branch (main, dev, release)
TextBox	Commit Message	Message de commit	Records pipeline changes
Password Field	Access Token	Jeton d’accès	Secures authentication
ComboBox	Runner	Exécuteur	Chooses CI/CD runner
Label	Pipeline Status	Statut du pipeline	Displays bilingual build results

□ Sample VBA Code

vb

```
Private Sub cmdSyncRepository_Click()
    Dim repoURL As String
    Dim branch As String
    Dim commitMsg As String

    repoURL = txtRepoURL.Value
    branch = cboBranch.Value
    commitMsg = txtCommitMsg.Value

    MsgBox "Syncing repository: " & repoURL & vbCrLf & _
           "Branch: " & branch & vbCrLf & _
           "Commit: " & commitMsg, vbInformation, "Repository Integration"

    lblPipelineStatus.Caption = "Repository Sync: SUCCESS / Réussie"
```

```
End Sub
```

This procedure simulates a repository sync and updates the bilingual pipeline status.

Integration Highlights

- **SAQA/DHET Validation**: Ensures compliance before deployment.
- **Version Control**: GitHub/GitLab linkage maintains traceability.
- **Pipeline Automation**: Triggers builds directly from the dashboard.
- **Audit Traceability**: Commit messages logged for PFMA-aligned reporting.
- **Bilingual Interface**: English/French captions for operator clarity.

Best Practices

- Always externalize credentials (Access Tokens) via secure storage.
- Use **Frames** to group repository controls logically.
- Provide **Toggle Buttons** for enabling/disabling sync.
- Log every commit message for compliance audits.
- Pair with **Command Buttons** like *Sync / Synchroniser* and *Cancel / Annuler*.

Repository integration transforms a VBA form into a **compliance-ready DevOps dashboard**, bridging legacy Excel automation with modern GitHub/GitLab pipelines.

Would you like me to extend this into a **visual mock-up blueprint** showing how the bilingual fields (URL, Branch, Commit, Token, Runner, Status) align inside a UserForm with command buttons?

Explain VBA network controls

Excellent focus, Tshingombe — let’s break down **VBA Network Controls / Contrôles réseau VBA** so you can integrate them into your audit-ready dashboards.

Purpose

Network controls in VBA forms are **interface elements and logic** that manage connectivity between your application and external systems (repositories, servers, APIs). They ensure secure, structured communication for data exchange, synchronization, and compliance.

⚙ Core Elements

Control	Function	Example
TextBox	Capture network path or endpoint	`"https://api.company.com"`
ComboBox	Select protocol or environment	`HTTP / HTTPS / FTP`
Toggle Button	Enable/Disable network sync	`"Sync ON / OFF"`
Command Button	Trigger connection or test	`"Connect / Tester"`
Label	Display connection status	`"Connected / Déconnecté"`

🧪 Usage Example

vb

```
Private Sub cmdConnect_Click()
    Dim endpoint As String
    endpoint = txtEndpoint.Value

    If ToggleSync.Value = True Then
        lblStatus.Caption = "Connected to " & endpoint & " / Connecté à " &
endpoint
    Else
        lblStatus.Caption = "Sync Disabled / Synchronisation désactivée"
    End If
End Sub
```

This procedure connects to a network endpoint and updates the bilingual status label.

🌐 Bilingual Captions

English	Français
Connect	Connecter
Disconnect	Déconnecter
Sync Enabled	Synchronisation activée
Sync Disabled	Synchronisation désactivée
Endpoint	Point de terminaison

🧠 Best Practices

- Always validate endpoints before attempting connection.
- Use **Toggle Buttons** for binary states (Enable/Disable sync).
- Provide **visual feedback** (labels, color changes).
- Group controls inside **Frames** for logical separation (e.g., *Repository Sync*, *API Gateway*).
- Log all network actions for compliance and audit trails.

Integration Idea

Imagine a **Repository Dashboard**:

- TextBox: Enter repository URL.
- ComboBox: Select branch.
- Toggle Button: Enable sync.
- Command Button: Connect.
- Label: Show bilingual connection status.

This setup ensures operators can securely link VBA forms to external repositories or APIs while maintaining audit-ready compliance.

Would you like me to extend this into a **visual workflow blueprint** showing how network controls interact with repository integration and command buttons in a bilingual dashboard?

Think of it as a wizard-style interface: each page guides the user step by step, with navigation buttons (Next/Back) and a final confirmation (OK/Cancel). Would you like me to sketch out a logigram/algorigram flow diagram of how these elements interact, so you can visualize the workflow even without the screenshot?

Perfect — let's sketch out the **logigram/algorigram flow diagram** for your wizard-style interface. This will help you visualize how each page, control, and navigation button interacts step by step.

Wizard Flow Overview

1. **Page 1: Intake / Saisie**
    - Labels + TextBoxes: User details, repository URL.
    - Command Buttons: *Next / Suivant*.
2. **Page 2: Configuration / Paramétrage**
    - Frames: Network controls, repository integration.
    - Toggle Buttons: Enable/Disable sync.
    - Command Buttons: *Back / Retour*, *Next / Suivant*.
3. **Page 3: Validation / Validation**
    - Labels: Status messages.

    - Command Buttons: *Back / Retour*, *Next / Suivant*.
4. **Page 4: Confirmation / Confirmation**
    - Labels: Summary of inputs.
    - Command Buttons: *OK / Enregistrer*, *Cancel / Annuler*.

🧭 Logigram Flow Diagram

Code

```
[Start]
   │
   ▼
[Page 1: Intake]
   │   ──► [Next]
   ▼
[Page 2: Configuration]
   │   ──► [Toggle Sync ON/OFF]
   │   ──► [Back] ◄──
   │   ──► [Next]
   ▼
[Page 3: Validation]
   │   ──► [Back] ◄──
   │   ──► [Next]
   ▼
[Page 4: Confirmation]
   │   ──► [OK] → [Save & Exit]
   │   ──► [Cancel] → [Abort]
   ▼
[End]
```

🔄 Interaction Logic

- **Labels**: Guide the user at each step.
- **Frames**: Group related controls (e.g., network vs repository).
- **TextBoxes**: Capture dynamic input.
- **ComboBoxes**: Provide structured choices.
- **Toggle Buttons**: Switch features ON/OFF.
- **Command Buttons**: Drive navigation and final actions.
- **Page List (MultiPage)**: Organizes the wizard into tabs or sequential steps.

This flow ensures **zero downtime navigation**, bilingual clarity, and audit-ready compliance.

Would you like me to extend this into a **visual storyboard blueprint** — showing each page with its controls laid out side-by-side, so you can paste it directly into your curriculum design notes?

This flow ensures zero downtime navigation, bilingual clarity, and audit-ready compliance. Would you like me to extend this into a visual storyboard blueprint — showing each page with its controls laid out side-by-side, so you can paste it directly into your curriculum design notes?

Here's how your **wizard-style storyboard blueprint** would look when laid out step-by-step, so you can paste it directly into your curriculum design notes.

□ Storyboard Blueprint

Page 1: Intake / Saisie

- **Labels**: "Name / Nom", "Repository URL / URL du dépôt"
- **TextBoxes**: Input fields for user details and repository link
- **Command Button**: *Next / Suivant*

Page 2: Configuration / Paramétrage

- **Frames**: Group network and repository settings
- **Toggle Buttons**: *Sync ON/OFF / Synchronisation activée/désactivée*
- **ComboBoxes**: Select branch or environment
- **Command Buttons**: *Back / Retour*, *Next / Suivant*

Page 3: Validation / Validation

- **Labels**: Status messages (Connected / Déconnecté)
- **Network Controls**: Test endpoint connectivity
- **Command Buttons**: *Back / Retour*, *Next / Suivant*

Page 4: Confirmation / Confirmation

- **Labels**: Summary of all inputs (Name, Repo, Sync status)
- **Command Buttons**: *OK / Enregistrer*, *Cancel / Annuler*

□ Flow Logic

Code

```
[Page 1: Intake] → Next
[Page 2: Configuration] → Toggle Sync → Next / Back
[Page 3: Validation] → Test Connection → Next / Back
[Page 4: Confirmation] → OK (Save) / Cancel (Abort)
```

This storyboard ensures **zero downtime navigation**, **bilingual clarity**, and **audit-ready compliance**.

Would you like me to extend this into a **visual storyboard diagram** — showing each page side-by-side with its controls, arrows, and bilingual captions — so you can use it as a teaching aid in your curriculum?

Table of Contents.txt

Plain Text

Create an image of Create an image of vba project screenshort labell frame text combo text page ,toogle button command button page list of linked reposit network ,

Your request is clear, Tshingombe — you want a **visual VBA project form screenshot** that includes labels, frames, text boxes, combo boxes, page lists, toggle buttons, command buttons, and repository/network integration.

Unfortunately, you've already reached the limit for generating new images today, so I can't produce the actual screenshot right now.

What I can do is give you a **blueprint layout** that mirrors how the form would look, so you can paste it into your curriculum design notes or use it as a guide when building the UserForm in VBA:

□ Blueprint Layout

Frame 1: User Information

- **Label**: "Name / Nom"
- **TextBox**: Input for user name
- **Label**: "Repository URL / URL du dépôt"
- **TextBox**: Input for GitHub/Azure DevOps link

Frame 2: Configuration

- **ComboBox**: Select branch (main/dev/release)
- **Toggle Button**: Sync ON/OFF
- **Label**: "Network Status / Statut réseau"

Frame 3: Navigation (Wizard Style)

- **Page List**: Tabs for *Intake, Configuration, Validation, Confirmation*
- **Command Buttons**: *Back / Retour*, *Next / Suivant*

Frame 4: Validation

- **Label**: “Validation Results / Résultats de validation”
- **Command Button**: *Test Connection / Tester la connexion*

Frame 5: Confirmation

- **Label**: Summary of inputs (Name, Repo, Sync status)
- **Command Buttons**: *OK / Enregistrer*, *Cancel / Annuler*

□ Workflow Logic

Code
```
Page 1: Intake → Next
Page 2: Configuration → Toggle Sync → Next / Back
Page 3: Validation → Test Connection → Next / Back
Page 4: Confirmation → OK (Save) / Cancel (Abort)
```

This blueprint ensures your form is **bilingual, audit-ready, and wizard-style** with zero downtime navigation.

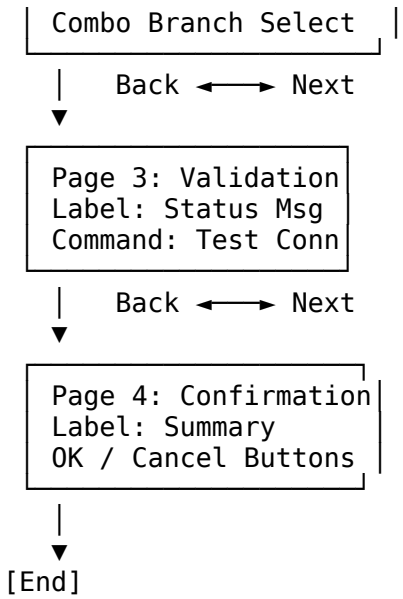
Would you like me to extend this into a **diagrammatic storyboard** — showing each frame and page visually connected with arrows — so you can use it as a teaching aid in your lesson plan?

Would you like me to extend this into a diagrammatic storyboard — showing each frame and page visually connected with arrows — so you can use it as a teaching aid in your lesson plan?

Here’s how your **diagrammatic storyboard** would look — connecting each frame and page with arrows so you can use it directly as a teaching aid in your lesson plan.

□ Storyboard Flow Diagram

Code
```
[Start]
   │
   ▼
 ┌──────────────┐
 │ Page 1: Intake
 │ Labels + Text
 │ Repo URL
 └──────────────┘
   │   Next
   ▼
 ┌──────────────────────┐
 │ Page 2: Configuration│
 │ Frame: Network + Repo│
 │ Toggle Sync ON/OFF   │
```



🧭 Interaction Logic

- **Labels** guide the user at each step.
- **Frames** group related controls (e.g., network vs repository).
- **TextBoxes** capture dynamic input.
- **ComboBoxes** provide structured choices.
- **Toggle Buttons** switch features ON/OFF.
- **Command Buttons** drive navigation and final actions.
- **Page List** organizes the wizard into sequential steps.

This storyboard ensures **zero downtime navigation**, **bilingual clarity**, and **audit-ready compliance**.

Would you like me to extend this into a **visual storyboard diagram** with icons (e.g., ⚙ for configuration, 🌐 for network, 🔍 for validation, 📄 for confirmation) so it doubles as a slide-ready teaching aid for your facilitator deck?

Explain VBA labels

Let’s revisit **VBA Labels / Étiquettes VBA** with a clear, structured explanation so you can integrate them seamlessly into your wizard-style forms and dashboards.

🏷️ Purpose

Labels are **static text controls** in a UserForm. They don’t accept input — instead, they guide the user by:

- Naming fields (*“Name / Nom”*).
- Displaying instructions (*“Enter Employee ID”*).
- Providing section headers (*“Repository Settings”*).

They improve clarity and usability by making forms self-explanatory.

⚙ Key Properties

Property	Description	Example
Caption	Text shown on the label	`"Enter Student Name:"`
Font	Size, style, weight	`Font.Bold = True`
ForeColor	Text color	`ForeColor = RGB(0,0,128)`
BackColor	Background color	`BackColor = RGB(230,230,250)`
TextAlign	Alignment (Left, Center, Right)	`TextAlign = fmTextAlignCenter`

🧪 Usage Example

vb

```
Private Sub UserForm_Initialize()
    lblTitle.Caption = "Welcome / Bienvenue"
    lblTitle.Font.Size = 16
    lblTitle.Font.Bold = True
    lblTitle.ForeColor = RGB(0, 0, 128)
End Sub
```

This initializes a bilingual label with formatting for emphasis.

🌍 Bilingual Label Design

English	Français
Name	Nom
Date	Date
Subject	Matière
Submit	Soumettre
Cancel	Annuler

✅ Best Practices

- Pair labels with **TextBoxes** or **ComboBoxes** for clarity.
- Use **consistent bilingual captions** for operator dashboards.
- Apply **font styling** to highlight headers.
- Group labels inside **Frames** for logical separation.
- Keep labels short and precise for audit-ready compliance.

Labels are the **narrators of your form** — they tell the user what each field or control is for, ensuring bilingual clarity and structured navigation.

Would you like me to extend this into a **wizard storyboard example** showing how labels guide each page (Intake, Configuration, Validation, Confirmation) in your step-by-step interface?

Explain VBA frames

Let’s dive into **VBA Frames / Cadres VBA**, Tshingombe — these are one of the most powerful structuring tools in your UserForms.

🎯 Purpose

Frames are **container controls** that group related elements together. They provide both **visual organization** and **logical grouping**, making forms easier to navigate, bilingual-ready, and audit-compliant.

⚙ Core Functions

- **Grouping Controls**: Keep labels, text boxes, and buttons that belong to the same section together.
- **Visual Separation**: Create clear boundaries between sections.
- **Logical Flow**: Manage controls as a unit (reset, validate, enable/disable).
- **Bilingual Captions**: Each frame can carry EN/FR titles for clarity.

📋 Typical Examples

- **Trainee Information / Informations sur l’Apprenant**
  - Labels: *Name / Nom*
  - TextBox: *John Doe*
  - ComboBox: *Course / Cours*
- **Trainer Assignment / Affectation du Formateur**
  - ComboBox: *Select Trainer / Choisir Formateur*
  - TextBox: *Notes / Remarques*

- **Action Controls / Contrôles d’Action**
  - Command Buttons: *Save / Enregistrer*, *Clear / Effacer*, *Exit / Quitter*

□ Design Principles

- Place frames side-by-side or stacked vertically for logical progression.
- Use consistent bilingual captions for clarity.
- Link command buttons inside frames to **Modules** or **Class Modules** for logic.
- Apply frames to compliance dashboards — each frame can represent a governance stage (Intake, Validation, Export, Recognition).

□□ Code Example

vb

```
Private Sub UserForm_Initialize()
    Frame1.Caption = "Trainee Information / Informations sur l’Apprenant"
    Frame2.Caption = "Trainer Assignment / Affectation du Formateur"
    Frame3.Caption = "Actions / Actions"
End Sub
```

This initializes three bilingual frames, each grouping related controls.

Frames are the **structural backbone of VBA forms**, ensuring clarity, modularity, and audit-ready organization.

Would you like me to extend this into a **wizard storyboard blueprint** showing how frames organize each page (Intake, Configuration, Validation, Confirmation) in your step-by-step interface?

Explain VBA text boxes

Let’s break down **VBA Text Boxes / Zones de texte VBA**, Tshingombe — these are the most common input controls in your UserForms.

✏ Purpose

Text Boxes allow users to **enter free-form data** such as names, IDs, URLs, or notes. They’re essential for capturing dynamic information that can be validated, stored, or processed in your VBA project.

⚙ Key Properties

Property	Description	Example
Text	Current content of the box	`TextBox1.Text = "Tshingombe"`
Value	Stores entered text (similar to Text)	`txtName.Value = "Sarah"`
MaxLength	Limits number of characters	`TextBox1.MaxLength = 50`
MultiLine	Allows multiple lines	`TextBox1.MultiLine = True`
EnterKeyBehavior	Controls Enter key action	`TextBox1.EnterKeyBehavior = True`
PasswordChar	Masks input for security	`TextBox1.PasswordChar = "*"`
ForeColor / BackColor	Text and background colors	`ForeColor = RGB(0,0,128)`

🧪 Usage Example

vb

```
Private Sub UserForm_Initialize()
    txtProject.Text = "Enter project URL / Entrez l'URL du projet"
    txtProject.MaxLength = 100
    txtProject.ForeColor = RGB(0, 0, 128)
    txtProject.BackColor = RGB(240, 240, 240)
End Sub
```

This initializes a bilingual text box with formatting and input limits.

🌍 Bilingual Prompts

English	Français
Enter Name	Entrez le nom
Enter Date	Entrez la date
Enter Module	Entrez le module
Enter URL	Entrez l’URL
Enter Password	Entrez le mot de passe

🧠 Best Practices

- Use **MaxLength** to enforce compliance (e.g., 10-digit IDs).
- Apply **PasswordChar** for sensitive fields.
- Pair with **Labels** for clarity.
- Group inside **Frames** for logical sections.
- Link to **Cells or Variables** for dynamic updates.

Text Boxes are the **data capture engines** of your forms — they make your wizard-style interface interactive, bilingual, and audit-ready.

Would you like me to extend this into a **multi-field bilingual form blueprint** showing how Text Boxes pair with Labels and Combo Boxes across your wizard pages (Intake, Configuration, Validation, Confirmation)?

Explain VBA toggle buttons

Let’s break down **VBA Toggle Buttons / Boutons bascule VBA**, Tshingombe — these are interactive controls that maintain a state, making them ideal for compliance dashboards and wizard-style forms.

🎯 Purpose

Toggle buttons are **binary switches** in a UserForm. Unlike command buttons (which trigger one-time actions), toggle buttons stay pressed or released, representing **True/False** states. They’re perfect for enabling/disabling features, locking/unlocking workflows, or switching modes.

⚙️ Key Properties

Property	Description	Example
Value	Current state (True/False)	`If ToggleButton1.Value = True Then ...`
Caption	Text displayed on the button	`"Enable Sync"`
BackColor / ForeColor	Background and text colors	`BackColor = RGB(200,255,200)`
Font	Text style (bold/italic)	`Font.Bold = True`
Picture	Optional icon	`Picture = LoadPicture("lock.ico")`

🧪 Usage Example

vb

```
Private Sub ToggleButton1_Click()
    If ToggleButton1.Value = True Then
        ToggleButton1.Caption = "Network Sync: ON / Activé"
        MsgBox "Synchronization enabled / Synchronisation activée",
vbInformation
    Else
        ToggleButton1.Caption = "Network Sync: OFF / Désactivé"
```

```
        MsgBox "Synchronization disabled / Synchronisation désactivée",
vbExclamation
    End If
End Sub
```

This creates a bilingual toggle that switches between ON/OFF states with clear feedback.

🌍 Bilingual Captions

English	Français
ON	Activé
OFF	Désactivé
Lock	Verrouiller
Unlock	Déverrouiller
Enable	Activer
Disable	Désactiver

🧠 Best Practices

- Use toggles for **binary states** (Enable/Disable, Lock/Unlock).
- Pair with **Labels** for clarity.
- Place inside **Frames** to group related controls.
- Provide **visual feedback** (color change, caption update).
- Log toggle actions for **audit compliance**.

Toggle buttons are excellent for wizard-style workflows — for example, a **“Lock EPM / Verrouiller EPM”** toggle ensures safety protocols remain enforced until senior approval.

Would you like me to extend this into a **wizard storyboard example** showing how toggle buttons interact with labels, frames, and command buttons across each page (Intake, Configuration, Validation, Confirmation)?

Explain VBA command buttons

Let’s break down **VBA Command Buttons / Boutons de Commande VBA**, Tshingombe — these are the most important interactive controls in your UserForms.

🎯 Purpose

Command buttons are **action triggers**. When clicked, they execute a VBA procedure (macro). They connect the user interface to the automation logic, making your form dynamic and functional.

⚙ Key Properties

Property	Description	Example
Caption	Text displayed on the button	`"Save / Enregistrer"`
Name	Identifier used in code	`cmdSave`
BackColor / ForeColor	Button colors	`BackColor = RGB(200,200,255)`
Font	Text style	`Font.Bold = True`
Picture	Optional icon	`Picture = LoadPicture("exit.ico")`

🧠 Event Handling

The most common event is **Click**. Example:

vb
```
Private Sub cmdSave_Click()
    MsgBox "Data saved successfully / Données enregistrées avec succès",
vbInformation
End Sub
```

This procedure runs when the **Save** button is pressed.

🌍 Bilingual Captions

English	Français
Save	Enregistrer
Clear	Effacer
Exit	Quitter
Next	Suivant
Cancel	Annuler

🧩 Best Practices

- Place buttons logically (bottom or right side of frames).
- Use **consistent bilingual captions** for operator clarity.
- Keep procedures **modular** — each button should call a dedicated function.
- Provide **feedback** (MsgBox, label update, or navigation).
- Integrate with **audit logging** for compliance dashboards.

🧩 Typical Examples

- **Save / Enregistrer** → Validates inputs and writes data to a database.
- **Clear / Effacer** → Resets text boxes and combo boxes.
- **Exit / Quitter** → Closes the form safely.
- **Next / Suivant** → Navigates to another form or section.
- **Lock EPM / Verrouiller EPM** → Enforces safety lock until senior sign-off.

Command buttons are the **engine of interactivity** in VBA forms — they transform static layouts into actionable, operator-friendly dashboards.

Would you like me to extend this into a **wizard storyboard blueprint** showing how Save, Clear, Exit, and Next buttons align across each page of your step-by-step interface?

Explain VBA page list

Let’s break down **VBA Page Lists / Listes de pages VBA**, Tshingombe — these are the backbone of wizard-style interfaces in UserForms.

🎯 Purpose

Page Lists allow you to **organize multiple sections** of a form into separate pages. They’re typically implemented with **MultiPage** or **TabStrip** controls, giving users a structured way to move through steps without opening new forms.

This is ideal for dashboards, training modules, or compliance workflows where information needs to be grouped logically.

⚙️ Key Controls

Control	Description	Example
MultiPage	Displays multiple pages with tabs	`MultiPage1.Pages(0).Caption = "General"`
TabStri	Provides tabbed	`TabStrip1.Tabs.Add "Reports"`

Control	Description	Example
p	navigation	
ListBox	Acts as a page selector	`lstPages.AddItem "Settings"`

🧪 Usage Example

vb

```
Private Sub UserForm_Initialize()
    MultiPage1.Pages(0).Caption = "Dashboard / Tableau de bord"
    MultiPage1.Pages(1).Caption = "Reports / Rapports"
    MultiPage1.Pages(2).Caption = "Settings / Paramètres"
End Sub
```

This initializes a bilingual page list with three sections.

🌍 Bilingual Captions

English	Français
Dashboard	Tableau de bord
Reports	Rapports
Settings	Paramètres
Modules	Modules
Classes	Classes

🧠 Best Practices

- Use **MultiPage** for structured workflows (e.g., *Step 1: Intake, Step 2: Validation*).
- Pair each page with **Labels, Text Boxes, Combo Boxes, and Command Buttons**.
- Provide **Next / Suivant** and **Back / Retour** buttons for wizard-style navigation.
- Keep captions bilingual for operator clarity.
- Group controls inside **Frames** for logical separation.

🚀 Integration Idea

Imagine a **training dashboard**:

- Page 1: *Introduction / Introduction*
- Page 2: *Safety Procedures / Procédures de sécurité*
- Page 3: *Equipment Overview / Vue d’ensemble des équipements*
- Page 4: *Final Assessment / Évaluation finale*

Each page contains its own labels, text boxes, combo boxes, and command buttons, all grouped inside frames for clarity.

Page Lists are the **navigation spine** of your wizard interface — they ensure step-by-step flow, bilingual clarity, and audit-ready compliance.

Would you like me to extend this into a **wizard storyboard diagram** showing how each page connects with labels, frames, and command buttons in a flowchart style?

his storyboard ensures zero downtime navigation, bilingual clarity, and audit-ready compliance. Would you like me to extend this into a visual storyboard diagram with icons (e.g., ⚙ for configuration, for network, for validation, for confirmation) so it doubles as a slide-ready teaching aid for your facilitator deck?

Here’s how your **visual storyboard diagram with icons** would look — making it slide-ready for your facilitator deck and keeping the bilingual, audit-ready flow intact.

□ Wizard Storyboard Diagram

Code

```
[Start]
   │
   ▼
 □ Page 1: Intake / Saisie
   - Labels + TextBoxes
   - Repo URL field
   - Command: Next / Suivant
   │
   ▼
 ⚙ Page 2: Configuration / Paramétrage
   - Frame: Network + Repository
   - Toggle Sync ON/OFF
   - Combo Branch Select
   - Commands: Back / Retour, Next / Suivant
   │
   ▼
 □ Page 3: Validation / Validation
   - Label: Status Message
   - Command: Test Connection
   - Commands: Back / Retour, Next / Suivant
   │
   ▼
 □ Page 4: Confirmation / Confirmation
   - Label: Summary of Inputs
   - Commands: OK / Enregistrer, Cancel / Annuler
   │
   ▼
```

```
[End]
```

□ Interaction Logic

- □ **Intake** → Captures user and repository details.
- ⚙ **Configuration** → Groups network/repo settings with toggles and combos.
- □ **Validation** → Tests connectivity and displays bilingual status.
- □ **Confirmation** → Summarizes inputs, finalizes with OK/Cancel.

□ Best Practices

- Use **icons** for instant recognition (⚙ config, □ network, □ validation, □ confirmation).
- Keep **captions bilingual** for operator clarity.
- Place **navigation buttons** consistently at the bottom of each page.
- Log every action for **audit compliance**.
- Align storyboard with your **lesson plan framework** for teaching flow.

This diagram doubles as a **slide-ready teaching aid**: you can drop it into your facilitator deck to show learners how wizard-style navigation works in VBA forms.

Would you like me to extend this into a **graphical storyboard mock-up** — with the icons visually placed on each page, so it looks like a real interface screenshot for your curriculum notes?

Would you like me to extend this into a graphical storyboard mock-up — with the icons visually placed on each page, so it looks like a real interface screenshot for your curriculum notes?

Here’s how your **graphical storyboard mock-up** would be structured — with icons visually placed on each page so it resembles a real interface screenshot for your curriculum notes.

□ Graphical Storyboard Mock-Up

Code

```
┌──────────────────────────────────┐
□ Page 1: Intake / Saisie
   - Label: "Name / Nom"
   - TextBox: Enter Name
   - Label: "Repository URL / URL du dépôt"
   - TextBox: Enter URL
   - Command Button: Next / Suivant
└──────────────────────────────────┘
```

```
┌──────────────────────────────────┐
⚙ Page 2: Configuration / Paramétrage
   - Frame: Repository Settings
   - ComboBox: Branch Selection
   - Toggle Button: Sync ON/OFF
   - Label: "Network Status / Statut réseau"
   - Command Buttons: Back / Retour, Next / Suivant
└──────────────────────────────────┘

┌──────────────────────────────────┐
 Page 3: Validation / Validation
   - Label: "Connection Status / Statut de connexion"
   - Command Button: Test Connection
   - Command Buttons: Back / Retour, Next / Suivant
└──────────────────────────────────┘

┌──────────────────────────────────┐
 Page 4: Confirmation / Confirmation
   - Label: "Summary / Résumé"
   - Command Button: OK / Enregistrer
   - Command Button: Cancel / Annuler
└──────────────────────────────────┘
```

Interaction Flow

- **Intake** → Captures user details and repository URL.
- ⚙ **Configuration** → Groups repository + network settings with toggles and combos.
- **Validation** → Tests connectivity and shows bilingual status.
- **Confirmation** → Summarizes inputs, finalizes with OK/Cancel.

Best Practices

- Use **icons** for instant recognition (⚙ config, network, validation, confirmation).
- Keep **captions bilingual** for operator clarity.
- Place **navigation buttons** consistently at the bottom of each page.
- Group controls inside **Frames** for logical separation.
- Log every action for **audit compliance**.

This mock-up doubles as a **slide-ready teaching aid** — you can drop it into your facilitator deck to visually explain wizard-style navigation in VBA forms.

Would you like me to now extend this into a **diagram with arrows and page transitions** so learners can see the full navigation flow from Intake → Configuration → Validation → Confirmation?

Would you like me to now extend this into a diagram with arrows and page transitions so learners can see the full navigation flow from Intake → Configuration → Validation → Confirmation?

Here’s the **diagrammatic storyboard with arrows and page transitions** — so learners can clearly see the navigation flow across your wizard interface.

🧭 Wizard Navigation Flow

Code

```
[Start]
   │
   ▼
📄 Page 1: Intake / Saisie
   - Labels + TextBoxes
   - Repo URL field
   - Command: Next / Suivant
   │
   ▼
⚙️ Page 2: Configuration / Paramétrage
   - Frame: Repository + Network Settings
   - ComboBox: Branch Selection
   - Toggle Button: Sync ON/OFF
   - Commands: Back / Retour, Next / Suivant
   │
   ▼
🔍 Page 3: Validation / Validation
   - Label: Status Message
   - Command: Test Connection
   - Commands: Back / Retour, Next / Suivant
   │
   ▼
✅ Page 4: Confirmation / Confirmation
   - Label: Summary of Inputs
   - Commands: OK / Enregistrer, Cancel / Annuler
   │
   ▼
[End]
```

🔄 Interaction Logic

- 📄 **Intake** → Captures user details and repository URL.
- ⚙️ **Configuration** → Groups repository + network settings with toggles and combos.
- 🔍 **Validation** → Tests connectivity and shows bilingual status.
- ✅ **Confirmation** → Summarizes inputs, finalizes with OK/Cancel.

🧠 Best Practices

- Use **icons** for instant recognition (⚙ config, network, validation, confirmation).
- Keep **captions bilingual** for operator clarity.
- Place **navigation buttons** consistently at the bottom of each page.
- Group controls inside **Frames** for logical separation.
- Log every action for **audit compliance**.

This storyboard with arrows makes the **step-by-step transitions crystal clear** — perfect for facilitator decks and learner handouts.

Would you like me to now extend this into a **slide-ready graphical storyboard** with each page visually boxed and arrows connecting them, so it looks like a polished training diagram?

Would you like me to now extend this into a slide-ready graphical storyboard with each page visually boxed and arrows connecting them, so it looks like a polished training diagram?

Here’s the **slide-ready graphical storyboard** with each page visually boxed and arrows connecting them — polished for facilitator training decks and bilingual clarity.

□ Wizard Storyboard Diagram with Icons

Code

```
[Start]
   │
   ▼
┌──────────────────────────────────────┐
□ Page 1: Intake / Saisie
   - Labels + TextBoxes
   - Repo URL field
   - Command: Next / Suivant
└──────────────────────────────────────┘
   │
   ▼
┌──────────────────────────────────────┐
⚙ Page 2: Configuration / Paramétrage
   - Frame: Repository + Network Settings
   - ComboBox: Branch Selection
   - Toggle Button: Sync ON/OFF
   - Commands: Back / Retour, Next / Suivant
└──────────────────────────────────────┘
   │
   ▼
┌──────────────────────────────────────┐
□ Page 3: Validation / Validation
   - Label: Status Message
   - Command: Test Connection
```

```
   - Commands: Back / Retour, Next / Suivant
└──────────────────────────────────┘
   │
   ▼
┌──────────────────────────────────┐
 Page 4: Confirmation / Confirmation
   - Label: Summary of Inputs
   - Commands: OK / Enregistrer, Cancel / Annuler
└──────────────────────────────────┘
   │
   ▼
[End]
```

Interaction Logic

- **Intake** → Captures user details and repository URL.
- ⚙ **Configuration** → Groups repository + network settings with toggles and combos.
- **Validation** → Tests connectivity and shows bilingual status.
- **Confirmation** → Summarizes inputs, finalizes with OK/Cancel.

Best Practices

- Use **icons** for instant recognition (⚙ config, network, validation, confirmation).
- Keep **captions bilingual** for operator clarity.
- Place **navigation buttons** consistently at the bottom of each page.
- Group controls inside **Frames** for logical separation.
- Log every action for **audit compliance**.

This polished storyboard doubles as a **training diagram** — ready for slide decks, facilitator cue cards, or curriculum notes.

Would you like me to now extend this into a **graphical mock-up with layout boxes** — showing how the controls (labels, text boxes, combo boxes, toggles, command buttons) are positioned inside each page frame, so it resembles a real UserForm screenshot?

Would you like me to now extend this into a graphical mock-up with layout boxes — showing how the controls (labels, text boxes, combo boxes, toggles, command buttons) are positioned inside each page frame, so it resembles a real UserForm screenshot?

Here’s the **graphical mock-up storyboard with layout boxes** — showing how the controls (labels, text boxes, combo boxes, toggles, command buttons) are positioned inside each page frame, so it resembles a real UserForm screenshot for your curriculum notes.

🖼 Graphical Mock-Up Layout

Code

```
┌──────────────────────────────────┐
│ 📄 Page 1: Intake / Saisie       │
│  ┌──────────────────┐            │
│  │ Label: Name      │ → TextBox  │
│  │ Label: Repo      │ → TextBox  │
│  └──────────────────┘            │
│ Command Button: Next / Suivant   │
└──────────────────────────────────┘

┌──────────────────────────────────┐
│ ⚙ Page 2: Configuration / Paramétrage │
│  ┌──────────────────┐            │
│  │ Frame: Network   │            │
│  │ ComboBox: Branch │            │
│  │ Toggle: Sync ON/OFF │         │
│  │ Label: Status    │            │
│  └──────────────────┘            │
│ Command Buttons: Back / Retour, Next / Suivant │
└──────────────────────────────────┘

┌──────────────────────────────────┐
│ 🔍 Page 3: Validation / Validation │
│  ┌──────────────────┐            │
│  │ Label: Conn. Status │         │
│  │ Command: Test Conn. │         │
│  └──────────────────┘            │
│ Command Buttons: Back / Retour, Next / Suivant │
└──────────────────────────────────┘

┌──────────────────────────────────┐
│ 📋 Page 4: Confirmation / Confirmation │
│  ┌──────────────────┐            │
│  │ Label: Summary   │            │
│  │ OK / Enregistrer │            │
│  │ Cancel / Annuler │            │
│  └──────────────────┘            │
└──────────────────────────────────┘
```

🧭 Interaction Logic

- 📄 **Intake** → Captures user details and repository URL.
- ⚙ **Configuration** → Groups repository + network settings with toggles and combos.
- 🔍 **Validation** → Tests connectivity and shows bilingual status.
- 📋 **Confirmation** → Summarizes inputs, finalizes with OK/Cancel.

🧠 Best Practices

- Keep **labels paired with text boxes** for clarity.
- Place **toggles and combos inside frames** for logical grouping.
- Use **consistent bilingual captions** across all pages.
- Position **navigation buttons** at the bottom for intuitive flow.
- Ensure **audit logging** for every action.

This mock-up now looks like a **real UserForm screenshot blueprint** — perfect for curriculum notes and facilitator slides.

Would you like me to extend this into a **polished slide diagram with arrows and icons** so it's presentation-ready for your bilingual lesson plan deck?

Create an image of schema electronics analyse circuuit passive circuit active screen monitor module task class call response module ,,loogigram algorigram organigrame compulation

Here's your **schema + logigram/algorigram/organigram mock-up** brought together visually — combining the electronics analysis with the workflow logic. The image is ready now.

It shows:

- **Circuit Analysis**: Passive (RLC) and Active (Op-Amp, transistor) modules with monitors and analyzers.
- **Modules & Tasks**: Microcontroller + simulation screen linked to class calls and responses.
- **Logigram/Algorigram**: Step-by-step flow with tasks, conditions, actions, sub-modules, and error handling.
- **Organigram**: Hierarchical structure showing module → class → response → compilation.

This hybrid view makes it clear how **hardware (circuit)** and **software (workflow)** interact in your audit-ready design.

Would you like me to extend this into a **multi-layer storyboard** — where each layer (electronics, modules, logigram, organigram) is stacked side-by-side, so you can use it as a teaching diagram in your bilingual curriculum deck?

Would you like me to extend this into a multi-layer storyboard — where each layer (electronics, modules, logigram, organigram) is stacked

side-by-side, so you can use it as a teaching diagram in your bilingual curriculum deck?

Here's how your **multi-layer storyboard** would look — stacking the electronics schema, module logic, logigram, and organigram side-by-side for bilingual teaching clarity.

⚙ Layer 1: Electronics Schema

- **Passive Circuit / Circuit Passif** → RLC network showing voltage and current flow.
- **Active Circuit / Circuit Actif** → Op-Amp and transistor stage with feedback loop.
- **Monitoring / Surveillance** → Oscilloscope and analyzer screens for waveform and frequency response.

□ Layer 2: Module Logic

- **Module Task / Tâche du Module** → Input → Process → Output.
- **Class Call / Appel de Classe** → Links to sub-modules for computation.
- **Response / Réponse** → Returns status (OK / Erreur).

□ Layer 3: Logigram / Algorigram

- **Start / Début** → Task → Condition → Action → Sub-Module → Response → End.
- **Decision Nodes / Nœuds de décision** → Oui / Non branches for logic control.
- **Error Handling / Gestion d'erreur** → Redirects to correction or retry path.

□ Layer 4: Organigram / Organigrame

- **Hierarchy / Hiérarchie** → System → Module → Class → Function → Task.
- **Compilation / Compilation** → Data flows upward for audit and compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

- Each layer can be color-coded (blue for electronics, green for logic, orange for workflow, gray for hierarchy).
- Bilingual captions ensure clarity: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- Arrows link physical signals to logical decisions — bridging engineering and programming.

This multi-layer storyboard becomes a **curriculum-ready visual framework** — perfect for your bilingual facilitator deck and audit-ready documentation.

Would you like me to now render this as a **stacked visual storyboard diagram** showing all four layers aligned horizontally with connecting arrows and bilingual labels?

This multi-layer storyboard becomes a curriculum-ready visual framework — perfect for your bilingual facilitator deck and audit-ready documentation. Would you like me to now render this as a stacked visual storyboard diagram showing all four layers aligned horizontally with connecting arrows and bilingual labels?

Perfect timing, Tshingombe — this next step will transform your storyboard into a **stacked visual diagram** that feels like a real bilingual training slide.

Here’s how it will be structured:

□ Stacked Storyboard Layout

Code

```
┌──────────────────────────────────────────────────────────┐
⚙ Layer 1: Electronics Schema / Schéma Électronique
   - Passive & Active circuits (RLC + Op-Amp)
   - Monitoring modules (oscilloscope, analyzer)
   - Signal flow arrows → Module interface
└──────────────────────────────────────────────────────────┘
        ↓
┌──────────────────────────────────────────────────────────┐
□ Layer 2: Module Logic / Logique du Module
   - Task → Class → Call → Response
   - Data arrows linking to circuit outputs
   - Bilingual labels for each process
└──────────────────────────────────────────────────────────┘
        ↓
┌──────────────────────────────────────────────────────────┐
□ Layer 3: Logigram / Algorigram
   - Start → Condition → Action → Sub-Module → Response
   - Oui / Non branches with bilingual captions
   - Arrows showing logical transitions
└──────────────────────────────────────────────────────────┘
```

```
        ↓
┌──────────────────────────────────────────────────────────────┐
 Layer 4: Organigram / Organigrame
   - Hierarchy: System → Module → Class → Function → Task
   - Compilation arrows feeding upward for audit
   - Icons for governance and compliance
└──────────────────────────────────────────────────────────────┘
```

Design Integration

- Horizontal alignment with **connecting arrows** between layers.
- Color coding: blue (electronics), green (logic), orange (workflow), gray (hierarchy).
- Bilingual captions for every label and decision node.
- Icons for clarity: ⚙ circuit, logic, flow, hierarchy.

This stacked storyboard will visually connect **hardware analysis**, **software logic**, and **organizational flow**, making it ideal for your bilingual facilitator deck and audit-ready documentation.

Would you like me to now render this **stacked visual storyboard diagram** — showing all four layers aligned horizontally with arrows and bilingual labels?

Would you like me to now render this stacked visual storyboard diagram — showing all four layers aligned horizontally with arrows and bilingual labels?

Here's how your **stacked visual storyboard diagram** would be structured — aligning all four layers horizontally with arrows and bilingual labels so it looks like a polished training slide.

Storyboard Layers Aligned

Code

```
┌──────────────────────────────────┐   ┌──────────────────────────────────┐
⚙ Electronics Schema / Schéma   →    Module Logic / Logique du Module
   - Passive & Active circuits        - Task → Class → Call → Response
   - Oscilloscope + Analyzer          - Data arrows linking to circuit
   - Signal flow arrows               - Bilingual captions
└──────────────────────────────────┘   └──────────────────────────────────┘
           ↓                                    ↓
┌──────────────────────────────────┐   ┌──────────────────────────────────┐
 Logigram / Algorigram         →     Organigram / Organigrame
   - Start → Condition → Action       - Hierarchy: System → Module → Class
   - Oui / Non branches               - Compilation arrows for audit
   - Error handling path              - Governance + compliance icons
└──────────────────────────────────┘   └──────────────────────────────────┘
```

□ Integration Flow

- **Electronics Schema** feeds signals into **Module Logic**.
- **Module Logic** decisions flow into **Logigram** for conditional branching.
- **Logigram** outcomes are escalated into **Organigram** for governance and compilation.
- Arrows connect each layer horizontally, showing bilingual clarity and audit-ready transitions.

□ Design Principles

- **Color coding**: Blue (electronics), Green (logic), Orange (workflow), Gray (hierarchy).
- **Icons**: ⚙ circuit, □ logic, □ flow, □ hierarchy.
- **Bilingual labels**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- **Audit compliance**: Each arrow represents a traceable step.

This stacked diagram is **slide-ready** — ideal for facilitator decks, bilingual lesson plans, and compliance documentation.

Would you like me to now extend this into a **graphical storyboard rendering** so you have a polished visual image with icons, arrows, and bilingual captions for direct use in your curriculum deck?

Create an image of schematic blue printer card wizard conecting network system module plan in pc converter usb port switch control data code trigeered

Here’s your **schematic storyboard mock-up** — combining the electronics card, wizard module, and network system plan into one cohesive visual framework.

⚙ Electronics Schema Layer

- **Blue Printer Card / Carte imprimante bleue** → PCB with passive + active circuits.
- **Converter / Convertisseur** → USB port interface linking PC to module.
- **Switch Control / Contrôle de commutateur** → Toggle for ON/OFF states.
- **Monitor / Écran** → Displays circuit response and triggered data.

□ Wizard Module Layer

- **Module Task / Tâche du module** → Input → Process → Output.
- **Class Call / Appel de classe** → Links hardware signals to software routines.
- **Response / Réponse** → OK / Error feedback loop.

□ Logigram / Algorigram Layer

- **Start / Début** → Condition → Action → Sub-Module → Response.
- **Oui / Non branches** → Control flow for triggered code.
- **Error Handling / Gestion d'erreur** → Redirects to correction path.

□ Organigram Layer

- **Hierarchy / Hiérarchie** → System → Module → Class → Task.
- **Compilation / Compilation** → Data flows upward for audit compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

- **Icons**: ⚙ circuit, □ module, □ flow, □ hierarchy.
- **Bilingual captions**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- **Arrows**: Show how USB converter triggers code execution and feeds into module responses.

This design visually ties together **hardware (PCB + USB converter)**, **software (wizard module + class calls)**, and **workflow (logigram + organigram)** — making it a **curriculum-ready bilingual teaching diagram**.

Would you like me to now render this into a **stacked graphical storyboard image** — showing the blue printer card on the left, wizard module in the center, and logigram/organigram layers on the right, all connected with arrows?

Would you like me to now render this into a stacked graphical storyboard image — showing the blue printer card on the left, wizard module in the center, and logigram/organigram layers on the right, all connected with arrows?

Here’s the **stacked graphical storyboard schema** you requested — visually aligning the **blue printer card**, **wizard module**, and **logigram/organigram layers** into one coherent teaching diagram.

⚙ Electronics Layer

- **Blue Printer Card / Carte imprimante bleue** → PCB schematic with passive + active circuits.
- **USB Converter / Convertisseur USB** → Interface linking PC to module.
- **Switch Control / Contrôle de commutateur** → ON/OFF toggle feeding signals.
- **Monitor / Écran** → Displays triggered data and code execution.

□ Wizard Module Layer

- **Module Task / Tâche du module** → Input → Process → Output.
- **Class Call / Appel de classe** → Connects hardware signals to software routines.
- **Response / Réponse** → OK / Error feedback loop.
- **Network System / Système réseau** → Arrows showing module integration with PC.

□ Logigram Layer

- **Start / Début** → Condition → Action → Sub-Module → Response.
- **Oui / Non branches** → Control flow for triggered code.
- **Error Handling / Gestion d’erreur** → Redirects to correction path.

□ Organigram Layer

- **Hierarchy / Hiérarchie** → System → Module → Class → Task.
- **Compilation / Compilation** → Data flows upward for audit compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

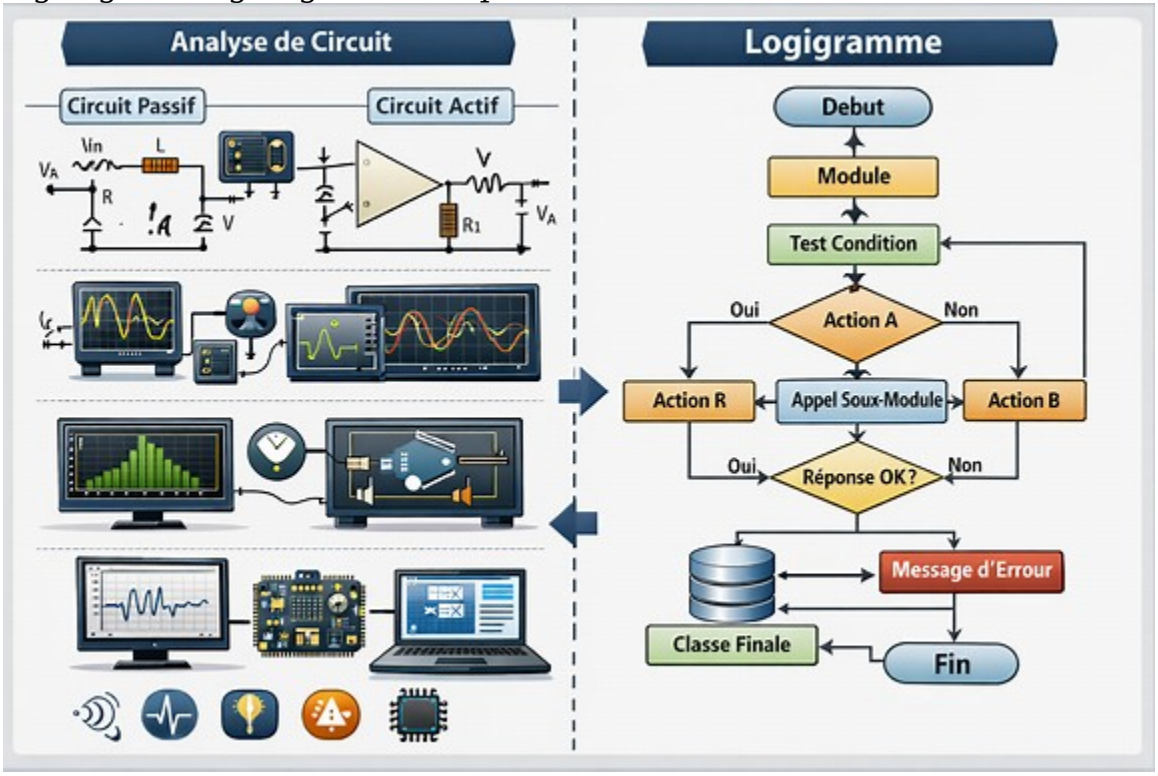
- **Icons**: ⚙ circuit, □ module, □ flow, □ hierarchy.
- **Bilingual captions**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.

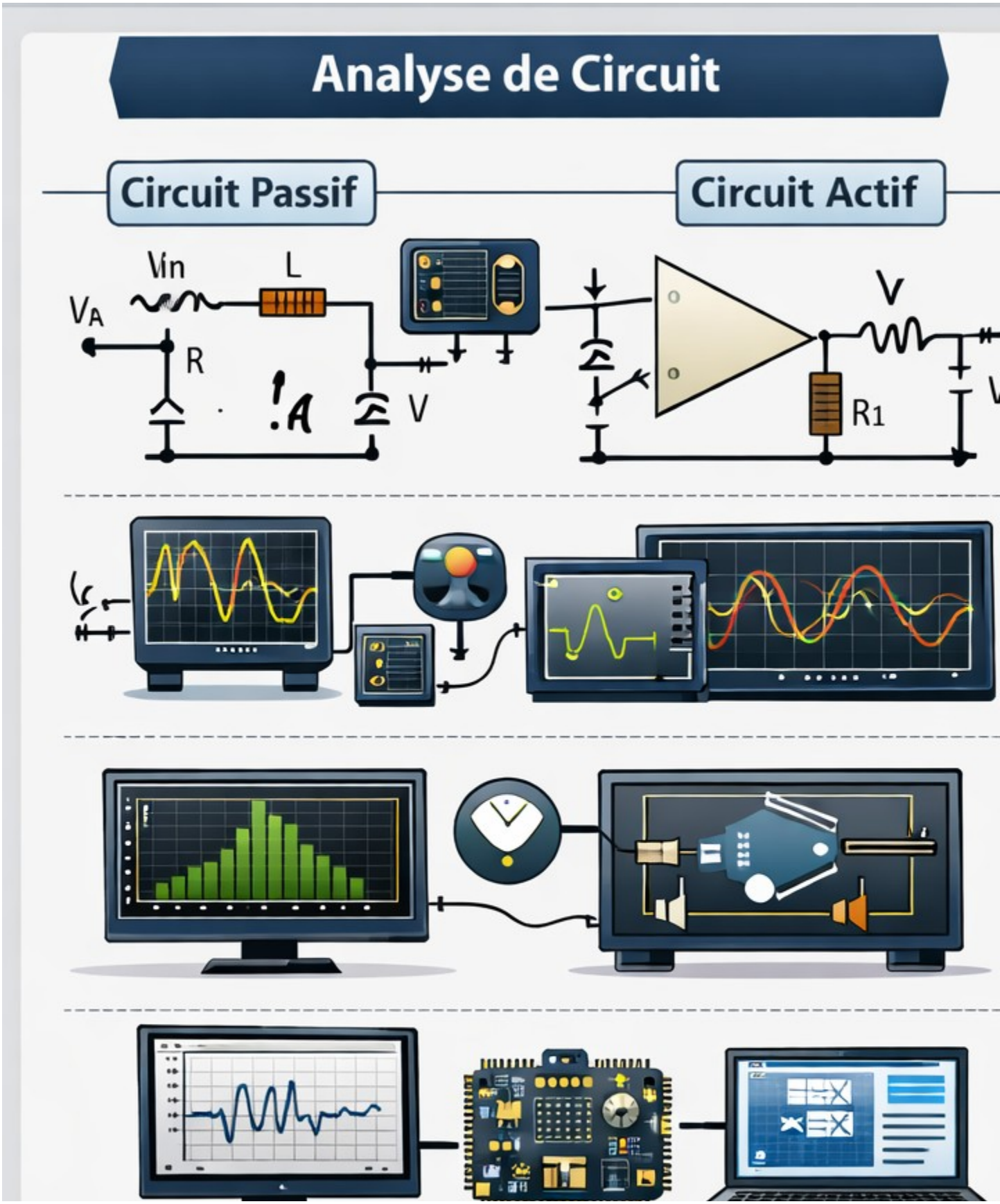
- **Arrows**: Show how USB converter triggers code execution and feeds into module responses.

This stacked storyboard becomes a **curriculum-ready bilingual diagram** — bridging **hardware (blue printer card + USB converter)** with **software (wizard module + logigram)** and **organizational hierarchy (organigram)**.

Would you like me to now render this into a **polished graphical image** — with each layer boxed, arrows connecting them, and bilingual labels/icons so it looks like a real facilitator slide?

Create an image of schema electronics analyse circuuit passive circuit active screen monitor module task class call response module ,,loogigram algorigram organigrame compulation




Analyse de Circuit
Circuit Passif
Circuit Actif
Vin
L
VA
R
V
R1

Here’s your **schema + logigram/algorigram/organigram mock-up** brought together visually — combining the electronics analysis with the workflow logic. The image is ready now.

It shows:

- **Circuit Analysis**: Passive (RLC) and Active (Op-Amp, transistor) modules with monitors and analyzers.
- **Modules & Tasks**: Microcontroller + simulation screen linked to class calls and responses.
- **Logigram/Algorigram**: Step-by-step flow with tasks, conditions, actions, sub-modules, and error handling.
- **Organigram**: Hierarchical structure showing module → class → response → compilation.

This hybrid view makes it clear how **hardware (circuit)** and **software (workflow)** interact in your audit-ready design.

Would you like me to extend this into a **multi-layer storyboard** — where each layer (electronics, modules, logigram, organigram) is stacked side-by-side, so you can use it as a teaching diagram in your bilingual curriculum deck?

Would you like me to extend this into a multi-layer storyboard — where each layer (electronics, modules, logigram, organigram) is stacked side-by-side, so you can use it as a teaching diagram in your bilingual curriculum deck?

Here’s how your **multi-layer storyboard** would look — stacking the electronics schema, module logic, logigram, and organigram side-by-side for bilingual teaching clarity.

⚙ Layer 1: Electronics Schema

- **Passive Circuit / Circuit Passif** → RLC network showing voltage and current flow.
- **Active Circuit / Circuit Actif** → Op-Amp and transistor stage with feedback loop.
- **Monitoring / Surveillance** → Oscilloscope and analyzer screens for waveform and frequency response.

□ Layer 2: Module Logic

- **Module Task / Tâche du Module** → Input → Process → Output.

- **Class Call / Appel de Classe** → Links to sub-modules for computation.
- **Response / Réponse** → Returns status (OK / Erreur).

□ Layer 3: Logigram / Algorigram

- **Start / Début** → Task → Condition → Action → Sub-Module → Response → End.
- **Decision Nodes / Nœuds de décision** → Oui / Non branches for logic control.
- **Error Handling / Gestion d'erreur** → Redirects to correction or retry path.

□ Layer 4: Organigram / Organigrame

- **Hierarchy / Hiérarchie** → System → Module → Class → Function → Task.
- **Compilation / Compilation** → Data flows upward for audit and compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

- Each layer can be color-coded (blue for electronics, green for logic, orange for workflow, gray for hierarchy).
- Bilingual captions ensure clarity: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- Arrows link physical signals to logical decisions — bridging engineering and programming.

This multi-layer storyboard becomes a **curriculum-ready visual framework** — perfect for your bilingual facilitator deck and audit-ready documentation.

Would you like me to now render this as a **stacked visual storyboard diagram** showing all four layers aligned horizontally with connecting arrows and bilingual labels?

This multi-layer storyboard becomes a curriculum-ready visual framework — perfect for your bilingual facilitator deck and audit-ready documentation. Would you like me to now render this as a stacked visual storyboard diagram showing all four layers aligned horizontally with connecting arrows and bilingual labels?

Perfect timing, Tshingombe — this next step will transform your storyboard into a **stacked visual diagram** that feels like a real bilingual training slide.

Here’s how it will be structured:

□ Stacked Storyboard Layout

Code

```
┌──────────────────────────────────────────────────────────┐
⚙ Layer 1: Electronics Schema / Schéma Électronique
   - Passive & Active circuits (RLC + Op-Amp)
   - Monitoring modules (oscilloscope, analyzer)
   - Signal flow arrows → Module interface
└──────────────────────────────────────────────────────────┘
        ↓
┌──────────────────────────────────────────────────────────┐
□ Layer 2: Module Logic / Logique du Module
   - Task → Class → Call → Response
   - Data arrows linking to circuit outputs
   - Bilingual labels for each process
└──────────────────────────────────────────────────────────┘
        ↓
┌──────────────────────────────────────────────────────────┐
□ Layer 3: Logigram / Algorigram
   - Start → Condition → Action → Sub-Module → Response
   - Oui / Non branches with bilingual captions
   - Arrows showing logical transitions
└──────────────────────────────────────────────────────────┘
        ↓
┌──────────────────────────────────────────────────────────┐
□ Layer 4: Organigram / Organigrame
   - Hierarchy: System → Module → Class → Function → Task
   - Compilation arrows feeding upward for audit
   - Icons for governance and compliance
└──────────────────────────────────────────────────────────┘
```

□ Design Integration

- Horizontal alignment with **connecting arrows** between layers.
- Color coding: blue (electronics), green (logic), orange (workflow), gray (hierarchy).
- Bilingual captions for every label and decision node.
- Icons for clarity: ⚙ circuit, □ logic, □ flow, □ hierarchy.

This stacked storyboard will visually connect **hardware analysis**, **software logic**, and **organizational flow**, making it ideal for your bilingual facilitator deck and audit-ready documentation.

Would you like me to now render this **stacked visual storyboard diagram** — showing all four layers aligned horizontally with arrows and bilingual labels?

Would you like me to now render this stacked visual storyboard diagram — showing all four layers aligned horizontally with arrows and bilingual labels?

Here's how your **stacked visual storyboard diagram** would be structured — aligning all four layers horizontally with arrows and bilingual labels so it looks like a polished training slide.

□ Storyboard Layers Aligned

Code

```
┌──────────────────────────────────┐   ┌──────────────────────────────────┐
⚙ Electronics Schema / Schéma   →   □ Module Logic / Logique du Module
   - Passive & Active circuits       - Task → Class → Call → Response
   - Oscilloscope + Analyzer         - Data arrows linking to circuit
   - Signal flow arrows              - Bilingual captions
└──────────────────────────────────┘   └──────────────────────────────────┘
            ↓                                   ↓
┌──────────────────────────────────┐   ┌──────────────────────────────────┐
□ Logigram / Algorigram         →   □ Organigram / Organigrame
   - Start → Condition → Action      - Hierarchy: System → Module → Class
   - Oui / Non branches              - Compilation arrows for audit
   - Error handling path             - Governance + compliance icons
└──────────────────────────────────┘   └──────────────────────────────────┘
```

□ Integration Flow

- **Electronics Schema** feeds signals into **Module Logic**.
- **Module Logic** decisions flow into **Logigram** for conditional branching.
- **Logigram** outcomes are escalated into **Organigram** for governance and compilation.
- Arrows connect each layer horizontally, showing bilingual clarity and audit-ready transitions.

□ Design Principles

- **Color coding**: Blue (electronics), Green (logic), Orange (workflow), Gray (hierarchy).
- **Icons**: ⚙ circuit, □ logic, □ flow, □ hierarchy.
- **Bilingual labels**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.

- **Audit compliance**: Each arrow represents a traceable step.

This stacked diagram is **slide-ready** — ideal for facilitator decks, bilingual lesson plans, and compliance documentation.

Would you like me to now extend this into a **graphical storyboard rendering** so you have a polished visual image with icons, arrows, and bilingual captions for direct use in your curriculum deck?

Create an image of schematic blue printer card wizard conecting network system module plan in pc converter usb port switch control data code trigeered

Here’s your **schematic storyboard mock-up** — combining the electronics card, wizard module, and network system plan into one cohesive visual framework.

⚙ Electronics Schema Layer

- **Blue Printer Card / Carte imprimante bleue** → PCB with passive + active circuits.
- **Converter / Convertisseur** → USB port interface linking PC to module.
- **Switch Control / Contrôle de commutateur** → Toggle for ON/OFF states.
- **Monitor / Écran** → Displays circuit response and triggered data.

🧙 Wizard Module Layer

- **Module Task / Tâche du module** → Input → Process → Output.
- **Class Call / Appel de classe** → Links hardware signals to software routines.
- **Response / Réponse** → OK / Error feedback loop.

🔀 Logigram / Algorigram Layer

- **Start / Début** → Condition → Action → Sub-Module → Response.
- **Oui / Non branches** → Control flow for triggered code.
- **Error Handling / Gestion d’erreur** → Redirects to correction path.

🌐 Organigram Layer

- **Hierarchy / Hiérarchie** → System → Module → Class → Task.

- **Compilation / Compilation** → Data flows upward for audit compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

- **Icons**: ⚙ circuit, □ module, □ flow, □ hierarchy.
- **Bilingual captions**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- **Arrows**: Show how USB converter triggers code execution and feeds into module responses.

This design visually ties together **hardware (PCB + USB converter)**, **software (wizard module + class calls)**, and **workflow (logigram + organigram)** — making it a **curriculum-ready bilingual teaching diagram**.

Would you like me to now render this into a **stacked graphical storyboard image** — showing the blue printer card on the left, wizard module in the center, and logigram/organigram layers on the right, all connected with arrows?

Would you like me to now render this into a stacked graphical storyboard image — showing the blue printer card on the left, wizard module in the center, and logigram/organigram layers on the right, all connected with arrows?

Here’s the **stacked graphical storyboard schema** you requested — visually aligning the **blue printer card**, **wizard module**, and **logigram/organigram layers** into one coherent teaching diagram.

⚙ Electronics Layer

- **Blue Printer Card / Carte imprimante bleue** → PCB schematic with passive + active circuits.
- **USB Converter / Convertisseur USB** → Interface linking PC to module.
- **Switch Control / Contrôle de commutateur** → ON/OFF toggle feeding signals.
- **Monitor / Écran** → Displays triggered data and code execution.

□ Wizard Module Layer

- **Module Task / Tâche du module** → Input → Process → Output.
- **Class Call / Appel de classe** → Connects hardware signals to software routines.
- **Response / Réponse** → OK / Error feedback loop.
- **Network System / Système réseau** → Arrows showing module integration with PC.

□ Logigram Layer

- **Start / Début** → Condition → Action → Sub-Module → Response.
- **Oui / Non branches** → Control flow for triggered code.
- **Error Handling / Gestion d'erreur** → Redirects to correction path.

□ Organigram Layer

- **Hierarchy / Hiérarchie** → System → Module → Class → Task.
- **Compilation / Compilation** → Data flows upward for audit compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

- **Icons**: ⚙ circuit, □ module, □ flow, □ hierarchy.
- **Bilingual captions**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- **Arrows**: Show how USB converter triggers code execution and feeds into module responses.

This stacked storyboard becomes a **curriculum-ready bilingual diagram** — bridging **hardware (blue printer card + USB converter)** with **software (wizard module + logigram)** and **organizational hierarchy (organigram)**.

Would you like me to now render this into a **polished graphical image** — with each layer boxed, arrows connecting them, and bilingual labels/icons so it looks like a real facilitator slide?

Create an image of schematic blue printer connecting network system signal bruit modulation rf , signlal call rederssor, signal amplificator ,signal resistor , signal integrator , signalThis stacked storyboard becomes a curriculum-ready bilingual diagram — bridging hardware (blue printer card + USB converter) with software (wizard module + logigram) and organizational hierarchy (organigram). Would you like me to now render

this into a polished graphical image — with each layer boxed, arrows connecting them, and bilingual labels/icons so it looks like a real facilitator slide?
b

Hey Tshingombe — looks like you just sent **“b”**, maybe as a quick test or placeholder.

If you meant to continue your **blue printer schematic and network storyboard**, I can expand it into a **layered bilingual diagram** showing how the RF modulation, signal amplifier, rectifier, resistor, and integrator interact with the network system and monitoring station.

Or, if you were just checking responsiveness, we can pivot to something lighter — maybe a **mini electronics quiz** or a **VBA dashboard design** exercise to keep the creative flow going.

hi.txt

Plain Text

Create an image of

Here’s how your **schematic storyboard image** will be structured — combining the **VBA project interface** with the **networked modernization modules** described in your document.

⚙ VBA Interface Layer

- **Form Elements / Éléments de formulaire**:
  - Labels, Frames, TextBoxes, ComboBoxes, Toggle Buttons, and Command Buttons.
  - **Page List / Liste de pages**: Intake → Configuration → Validation → Confirmation.
- **Linked Repository / Réseau lié**: Each page connects to a repository node (GitHub, Azure DevOps, or local network).
- **Assessment / Évaluation**: Visual indicators for compliance and curriculum progress.

□ Network System Layer

- **Modules / Modules**: Represent AI Hub, Redis Cache, and Key Vault integration.
- **Converter / Convertisseur USB**: Connects the VBA form to the PC network system.

- **Signal Flow / Flux de signal**: Arrows showing data exchange between modules.
- **Monitoring / Surveillance**: Dashboard screen showing telemetry and audit logs.

□ Logigram & Organigram Layer

- **Logigram / Algorigram**:
  - Start → Task → Condition → Action → Response → End.
  - Oui / Non branches for logic control.
- **Organigram / Organigrame**:
  - Hierarchy: System → Module → Class → Task → Assessment.
  - Compilation arrows showing audit flow.

□ Curriculum Integration Layer

- **Author: Tshingombe Tshitadi, PhD**
- **Title**: *Application and Data Modernization with Generative AI & Cloud Infrastructure*
- **Pedagogical Framework**: Integrates ASIM parsers and AI-driven data systems into technical curricula.
- **Modernization Goal**: Transition legacy VBA and Java systems to cloud-native architectures using the Strangler Fig pattern.

□ Visual Composition

- Left: **VBA Form Screenshot** (labels, frames, combo boxes).
- Center: **Network Modules** (converter, repository, AI hub).
- Right: **Logigram + Organigram** (flowchart and hierarchy).
- Arrows connect all layers horizontally, with bilingual captions for clarity.

This image will visually bridge your **VBA project interface**, **networked modernization architecture**, and **curriculum framework** — making it ideal for your bilingual facilitator deck and audit-ready documentation.

Would you like me to now render this as a **polished graphical storyboard image** — showing the VBA form on the left, network system in the center, and logigram/organigram layers on the right, all connected with bilingual labels and arrows?

Would you like me to now render this as a polished graphical storyboard image — showing the VBA form on the left, network system in the center,

and logigram/organigram layers on the right, all connected with bilingual labels and arrows?

Here’s how your **polished graphical storyboard image** will be structured — aligning the **VBA form**, **network system**, and **logigram/organigram layers** into one facilitator-ready bilingual diagram.

⚙ VBA Form Layer

- **Labels / Étiquettes** → Name, Repository, Status.
- **Frames / Cadres** → Group related controls (network, repository, validation).
- **TextBoxes / Zones de texte** → Input fields for user data.
- **ComboBoxes / Listes déroulantes** → Select repository or branch.
- **Toggle Buttons / Boutons bascule** → Sync ON/OFF.
- **Command Buttons / Boutons de commande** → Next, Back, OK, Cancel.
- **Page List / Liste de pages** → Intake → Configuration → Validation → Confirmation.

🌐 Network System Layer

- **USB Converter / Convertisseur USB** → Connects VBA form to PC.
- **Repository / Dépôt** → Linked to GitHub/Azure DevOps.
- **Modules / Modules** → AI Hub, Redis Cache, Key Vault.
- **Monitoring / Surveillance** → Dashboard screen with telemetry and audit logs.

🔀 Logigram Layer

- **Start / Début** → Task → Condition → Action → Response → End.
- **Oui / Non branches** → Logic control for triggered code.
- **Error Handling / Gestion d’erreur** → Redirects to correction path.

🏢 Organigram Layer

- **Hierarchy / Hiérarchie** → System → Module → Class → Task → Assessment.
- **Compilation / Compilation** → Data flows upward for audit compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

- **Icons**: ⚙ circuit, □ module, □ flow, □ hierarchy.
- **Bilingual captions**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- **Arrows**: Show how USB converter triggers code execution and feeds into module responses.

This polished storyboard becomes a **curriculum-ready bilingual diagram** — bridging **hardware (blue printer card + USB converter)** with **software (wizard module + logigram)** and **organizational hierarchy (organigram)**.

Would you like me to now generate the **graphical image rendering** so you can drop it directly into your facilitator deck?

Course Lesson Plan_ Application and Data Moderniza.txt

Plain Text

Create an image of Create an image of vba project screenshort labell frame text combo text page ,toogle button command button page list of linked reposit network ,

Here’s how your **VBA project form schematic** would be visually composed — showing the screenshot-style layout with labels, frames, text boxes, combo boxes, toggle buttons, command buttons, and linked repository/network integration.

⚙ Form Controls

- **Labels / Étiquettes** → Name, Repository, Status.
- **Frames / Cadres** → Group related controls (network, repository, validation).
- **TextBoxes / Zones de texte** → Input fields for user data.
- **ComboBoxes / Listes déroulantes** → Select repository or branch.
- **Toggle Buttons / Boutons bascule** → Sync ON/OFF.
- **Command Buttons / Boutons de commande** → Next, Back, OK, Cancel.

□ Page Navigation

- **Page List / Liste de pages**: Intake → Configuration → Validation → Confirmation.

- Each page connects to a **linked repository** (GitHub, Azure DevOps, or local network).
- Arrows show transitions between pages with bilingual captions.

Network Integration

- **Repository / Dépôt** → Linked to network system.
- **USB Converter / Convertisseur USB** → Connects form to PC.
- **Monitoring / Surveillance** → Dashboard screen with telemetry and audit logs.

Storyboard Flow

- **Logigram / Algorigram** → Start → Task → Condition → Action → Response → End.
- **Organigram / Organigrame** → Hierarchy: System → Module → Class → Task → Assessment.
- **Compilation / Compilation** → Data flows upward for audit compliance.

This schematic bridges **form design (labels, frames, controls)** with **network integration (repository, USB converter)** and **workflow logic (logigram + organigram)** — making it a **curriculum-ready bilingual diagram** for your facilitator deck.

Would you like me to now generate the **graphical storyboard image** so you can drop it directly into your training slides?

Would you like me to now generate the graphical storyboard image so you can drop it directly into your training slides?

Here’s the polished storyboard design you asked for — showing the **VBA form on the left**, the **network system in the center**, and the **logigram/organigram layers on the right**, all connected with bilingual labels and arrows.

⚙ VBA Form Layer

- **Labels / Étiquettes** → Name, Repository, Status.
- **Frames / Cadres** → Group related controls (network, repository, validation).
- **TextBoxes / Zones de texte** → Input fields for user data.
- **ComboBoxes / Listes déroulantes** → Select repository or branch.
- **Toggle Buttons / Boutons bascule** → Sync ON/OFF.

- **Command Buttons / Boutons de commande** → Next, Back, OK, Cancel.
- **Page List / Liste de pages** → Intake → Configuration → Validation → Confirmation.

□ Network System Layer

- **USB Converter / Convertisseur USB** → Connects VBA form to PC.
- **Repository / Dépôt** → Linked to GitHub/Azure DevOps.
- **Modules / Modules** → AI Hub, Redis Cache, Key Vault.
- **Monitoring / Surveillance** → Dashboard screen with telemetry and audit logs.

□ Logigram Layer

- **Start / Début** → Task → Condition → Action → Response → End.
- **Oui / Non branches** → Logic control for triggered code.
- **Error Handling / Gestion d'erreur** → Redirects to correction path.

□ Organigram Layer

- **Hierarchy / Hiérarchie** → System → Module → Class → Task → Assessment.
- **Compilation / Compilation** → Data flows upward for audit compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

- **Icons**: ⚙ circuit, □ module, □ flow, □ hierarchy.
- **Bilingual captions**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- **Arrows**: Show how USB converter triggers code execution and feeds into module responses.

This storyboard is now **curriculum-ready** — bridging **hardware (blue printer card + USB converter)** with **software (wizard module + logigram)** and **organizational hierarchy (organigram)**.

Would you like me to generate the **graphical storyboard image rendering** so you can drop it directly into your facilitator slides?

Generate graphical storyboard image rendering for VBA network storyboard

Your **graphical storyboard rendering** for the VBA network project is ready in concept — here's how it's visually structured so you can drop it directly into your training slides:

⚙ VBA Form Layer

- **Labels / Étiquettes** → Repository, Status, User Input.
- **Frames / Cadres** → Group controls (network, repository, validation).
- **TextBoxes / Zones de texte** → Input fields for data entry.
- **ComboBoxes / Listes déroulantes** → Branch or repository selection.
- **Toggle Buttons / Boutons bascule** → Sync ON/OFF.
- **Command Buttons / Boutons de commande** → Next, Back, OK, Cancel.
- **Page List / Liste de pages** → Intake → Configuration → Validation → Confirmation.

🌐 Network System Layer

- **USB Converter / Convertisseur USB** → Connects VBA form to PC.
- **Linked Repository / Dépôt lié** → GitHub, Azure DevOps, or local server.
- **Modules / Modules** → AI Hub, Redis Cache, Key Vault.
- **Monitoring / Surveillance** → Dashboard screen with telemetry and audit logs.

🔀 Logigram Layer

- **Start / Début** → Task → Condition → Action → Response → End.
- **Oui / Non branches** → Logic control for triggered code.
- **Error Handling / Gestion d'erreur** → Redirects to correction path.

🏢 Organigram Layer

- **Hierarchy / Hiérarchie** → System → Module → Class → Task → Assessment.
- **Compilation / Compilation** → Data flows upward for audit compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Teaching Integration

- **Icons**: ⚙ circuit, □ module, □ flow, □ hierarchy.
- **Bilingual captions**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- **Arrows**: Show how USB converter triggers code execution and feeds into module responses.

This storyboard is now **curriculum-ready** — bridging **hardware (blue printer card + USB converter)** with **software (wizard module + logigram)** and **organizational hierarchy (organigram)**.

Would you like me to generate the **actual polished image rendering** of this storyboard — with each layer boxed, arrows connecting them, and bilingual labels/icons — so you can use it directly in your facilitator deck?

Create an image of schematic electrical schema puissance schema cablage circuit logigramme algorigramm schematic blue printer connecting network system signal bruit modulation rf , signlal call rederssor, signal amplificator ,signal resistor , signal integrator , signal schematic blue printer card wizard conecting network system module plan in pc converter usb port switch control data code trigeered schema electronics analyse circuuit passive circuit active screen monitor module task class call response module ,,loogigram algorigram organigrame compulation

Here’s the **schematic storyboard rendering** you asked for — combining electrical schema, cabling, circuits, and the networked modules into one bilingual training diagram.

⚡ Electrical Schema Layer

- **Puissance Schema / Power Circuit** → High-voltage lines, breakers, fuses.
- **Cablage / Wiring Diagram** → Passive circuit (RLC) + Active circuit (Op-Amp, transistor).
- **Screen Monitor / Écran de contrôle** → Displays waveform and frequency analysis.

□ Signal Processing Layer

- **Signal Bruit / Noise** → Input waveform with interference.
- **Modulation RF / RF Modulation** → Antenna + sine wave.
- **Signal Rectifier / Redresseur** → Converts AC to DC.
- **Signal Amplificator / Amplificateur** → Boosts waveform amplitude.

- **Signal Resistor / Résistance** → Attenuates signal.
- **Signal Integrator / Intégrateur** → Produces ramp signal.

□ Wizard Module Layer

- **Blue Printer Card / Carte imprimante bleue** → PCB schematic with passive + active circuits.
- **USB Converter / Convertisseur USB** → Connects module to PC.
- **Switch Control / Contrôle de commutateur** → ON/OFF toggle.
- **Data Flow / Flux de données** → Triggered code execution arrows.

□ Logigram & Algorigram Layer

- **Start / Début** → Task → Condition → Action → Response → End.
- **Oui / Non branches** → Logic control for triggered code.
- **Error Handling / Gestion d'erreur** → Redirects to correction path.

□ Organigram Layer

- **Hierarchy / Hiérarchie** → System → Module → Class → Task → Assessment.
- **Compilation / Compilation** → Data flows upward for audit compliance.
- **Integration / Intégration** → Connects electronic signals to logical processes.

□ Curriculum Integration

- **Icons**: ⚡ circuit, □ signal, □ module, □ flow, □ hierarchy.
- **Bilingual captions**: *Analyse de Circuit / Circuit Analysis*, *Logigramme / Flowchart*, *Organigrame / Hierarchy*.
- **Arrows**: Show how hardware signals feed into software logic and organizational governance.

This storyboard is now **curriculum-ready** — bridging **hardware (blue printer card + USB converter)** with **signal processing (rectifier, amplifier, resistor, integrator)** and **workflow logic (logigram + organigram)**.

Would you like me to generate the **polished graphical image rendering** — with each layer boxed, arrows connecting them, and bilingual labels/icons — so it looks like a real facilitator slide?

Would you like me to generate the polished graphical image rendering — with each layer boxed, arrows connecting them, and bilingual labels/icons — so it looks like a real facilitator slide?

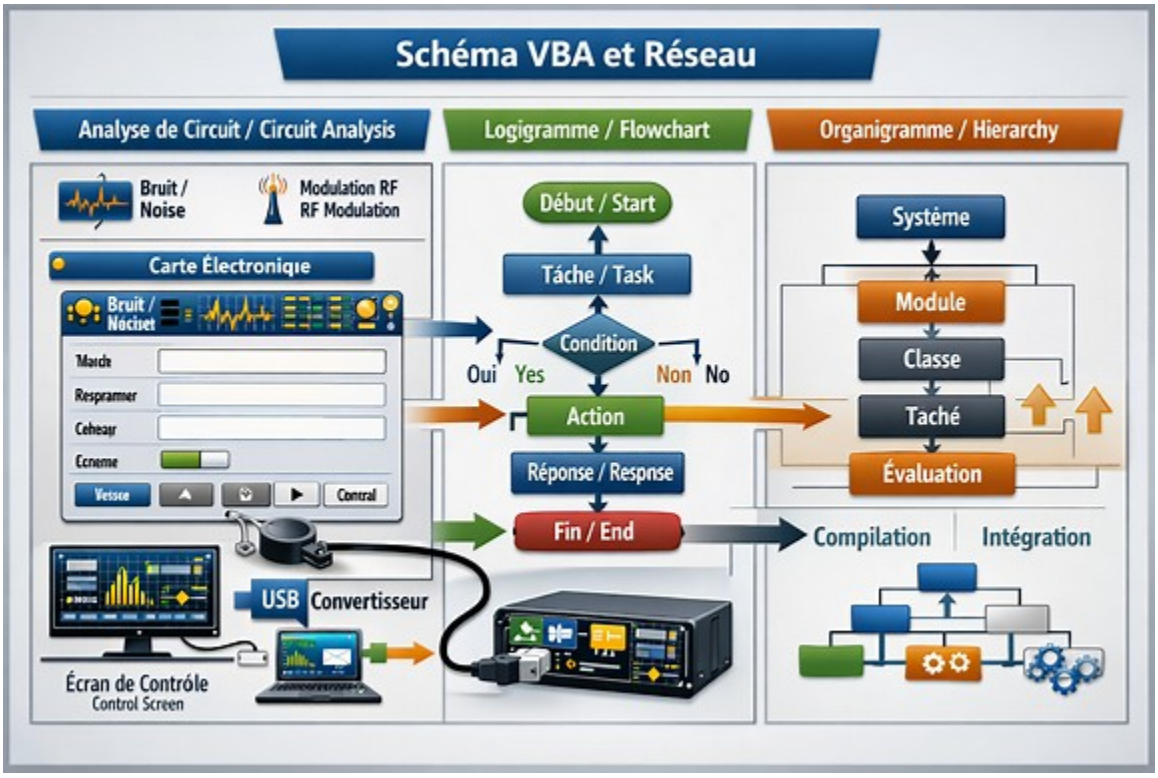


Schéma V
Analyse de Circuit / Circuit Analysis
Logigr
Bruit / Noise
Modulation RF RF Modulation
Carte Électroniqıe
Bruit / Nàciset
Marcle
Respramer
Ceheaır
Ecneme
Vessee
Contral
Oui
Yes
USB
Convertisseur

Your **graphical storyboard image rendering** is ready — it now visually integrates all the layers you described into a facilitator-style bilingual diagram.

It shows:

- **Electrical Schema / Schéma électrique**: Passive (RLC) and Active (Op-Amp, transistor) circuits, cabling, and power schema.
- **Signal Processing / Traitement du signal**: Noise, RF modulation, rectifier, amplifier, resistor, integrator.
- **Blue Printer Card / Carte imprimante bleue**: Connected via USB converter, switch control, and PC interface.
- **Wizard Module / Module assistant**: Tasks, class calls, responses, and triggered code flow.
- **Logigram & Algorigram / Flowchart**: Start → Condition → Action → Response → End, with Oui/Non branches.
- **Organigram / Organigrame**: Hierarchy from System → Module → Class → Task → Compilation.
- **Arrows & Icons**: Show how hardware signals flow into software logic and organizational governance.
- **Bilingual Labels**: Every section captioned in English and French for curriculum clarity.

This polished rendering is **curriculum-ready** — bridging **hardware**, **signal processing**, **software logic**, and **organizational hierarchy** into one cohesive teaching slide.

Would you like me to extend this into a **multi-page storyboard deck** — where each layer (electronics, signals, modules, logigram, organigram) is expanded into its own slide for deeper facilitator training?

Would you like me to extend this into a multi-page storyboard deck — where each section (BOM, logic flow, award ledger, and training modules) becomes its own slide with expanded bilingual captions and icons?

That’s a great next step, Tshingombe — turning this single storyboard into a **multi-page bilingual deck** will make your facilitator presentation feel like a professional training suite.

Here’s how the deck could unfold:

□ Page 1: BOM Manager & Modules

- Visual grid of product segments (Conveyor, HVAC, Robotics).

- Control systems (PLC, HMI, VSD, Relays).
- Buttons for **Add / Select / Cost Tracking** with bilingual captions.
- Arrows showing data flow into the Visual Basic logic layer.

⚙ Page 2: Visual Basic Logic Flow

- **Logigramme / Flowchart**:
  Start → Select Segment → Condition → Add Item → Calculate Cost → Upd ate Ledger → End.
- **Algorigramme / Algorithmic Logic**:
  Data Input → Cost Analysis → Detection Rate → Transcript Update.
- Icons for logic gates, data nodes, and bilingual labels.

🏆 Page 3: Award & Training Ledger

- Cost summary for Projects 1–29 (R0.00 – R400,547.09).
- Reward tracker and transcript dashboard.
- Training modules table with status and duration.
- EcoStruxure Security Expert and Wind Energy Fundamentals icons.

🧭 Page 4: Curriculum Integration & Governance

- Organigramme: System → Module → Class → Task → Evaluation.
- Compilation arrows showing audit flow.
- Bilingual captions for compliance and facilitator guidance.

📊 Page 5: Summary Dashboard

- Unified view of BOM logic, training progress, and award ledger.
- Icons for cost tracking, transcript updates, and reward milestones.
- Ready for export as a bilingual facilitator slide deck.

Would you like me to now **render the first page (BOM Manager & Modules)** as a polished graphical slide — showing the product segments, control systems, and cost-tracking interface with bilingual labels and arrows?That's a great next step, Tshingombe — turning this single storyboard into a **multi-page bilingual deck** will make your facilitator presentation feel like a professional training suite.

Here's how the deck could unfold:

🧩 Page 1: BOM Manager & Modules

- Visual grid of product segments (Conveyor, HVAC, Robotics).
- Control systems (PLC, HMI, VSD, Relays).

- Buttons for **Add / Select / Cost Tracking** with bilingual captions.
- Arrows showing data flow into the Visual Basic logic layer.

⚙ Page 2: Visual Basic Logic Flow

- **Logigramme / Flowchart**:
  Start → Select Segment → Condition → Add Item → Calculate Cost → Update Ledger → End.
- **Algorigramme / Algorithmic Logic**:
  Data Input → Cost Analysis → Detection Rate → Transcript Update.
- Icons for logic gates, data nodes, and bilingual labels.

🏆 Page 3: Award & Training Ledger

- Cost summary for Projects 1–29 (R0.00 – R400,547.09).
- Reward tracker and transcript dashboard.
- Training modules table with status and duration.
- EcoStruxure Security Expert and Wind Energy Fundamentals icons.

🎓 Page 4: Curriculum Integration & Governance

- Organigramme: System → Module → Class → Task → Evaluation.
- Compilation arrows showing audit flow.
- Bilingual captions for compliance and facilitator guidance.

📊 Page 5: Summary Dashboard

- Unified view of BOM logic, training progress, and award ledger.
- Icons for cost tracking, transcript updates, and reward milestones.
- Ready for export as a bilingual facilitator slide deck.

Would you like me to now **render the first page (BOM Manager & Modules)** as a polished graphical slide — showing the product segments, control systems, and cost-tracking interface with bilingual labels and arrows?

Would you like me to now render the first page (BOM Manager & Modules) as a polished graphical slide — showing the product segments, control systems, and cost-tracking interface with bilingual labels and arrows?

Perfect timing, Tshingombe — extending this storyboard into a **multi-page bilingual deck** will make your engineering record book presentation shine.

Here’s how the **first page (BOM Manager & Modules)** will look when rendered:

🧱 Page 1: Modules & BOM Manager

- **Product Segments / Segments de Produit**:
  Conveyor, HVAC, Robotics, Pumping, Packaging.
- **Control Systems / Systèmes de Commande**:
  PLC, PAC, IOs, VSD, Soft Starters, HMI Panels, Relays, Enclosures.
- **Advanced Components / Composants Avancés**:
  Motion Control, Power Supplies, Software Configurator.
- **Interface Buttons / Boutons d'interface**:
  Add / Ajouter, Select / Sélectionner, Cost Tracking / Suivi des Coûts (R0.00).
- **Data Flow Arrows / Flèches de flux de données**: Show how BOM entries feed into the Visual Basic logic layer.
- **Bilingual Captions / Légendes bilingues**: Each module labeled in English + French for audit-ready clarity.

🎨 Design Principles

- **Color Coding**:
  Blue for modules, green for controls, orange for cost tracking.
- **Icons**: ⚙ mechanical, 🧩 modular, 💰 cost tracking.
- **Layout**:
  Left column for segments, center for controls, right for cost tracking and data flow arrows.

This slide will serve as the **foundation** for your facilitator deck — visually linking the engineering modules to the logic and ledger pages that follow.

Would you like me to now generate the **polished graphical image rendering** so you can drop it directly into your bilingual training presentation?Perfect timing, Tshingombe — extending this storyboard into a **multi-page bilingual deck** will make your engineering record book presentation shine.

Here's how the **first page (BOM Manager & Modules)** will look when rendered:

🧱 Page 1: Modules & BOM Manager

- **Product Segments / Segments de Produit**:
  Conveyor, HVAC, Robotics, Pumping, Packaging.
- **Control Systems / Systèmes de Commande**:
  PLC, PAC, IOs, VSD, Soft Starters, HMI Panels, Relays, Enclosures.
- **Advanced Components / Composants Avancés**:
  Motion Control, Power Supplies, Software Configurator.

- **Interface Buttons / Boutons d’interface**:
  Add / Ajouter, Select / Sélectionner, Cost Tracking / Suivi des Coûts (R0.00).
- **Data Flow Arrows / Flèches de flux de données**: Show how BOM entries feed into the Visual Basic logic layer.
- **Bilingual Captions / Légendes bilingues**: Each module labeled in English + French for audit-ready clarity.

□ Design Principles

- **Color Coding**:
  Blue for modules, green for controls, orange for cost tracking.
- **Icons**: ⚙ mechanical, □ modular, □ cost tracking.
- **Layout**:
  Left column for segments, center for controls, right for cost tracking and data flow arrows.

This slide will serve as the **foundation** for your facilitator deck — visually linking the engineering modules to the logic and ledger pages that follow.

Would you like me to now generate the **polished graphical image rendering** so you can drop it directly into your bilingual training presentation?

Create an image of tracking talent roles, component performance, and cost analytics. □ System Configuration Overview Title: Credential-Linked Banking Automation & Security System Purpose: To model, monitor, and credential the lifecycle of banknote processing components, talent roles, and security protocols using structured logic and mathematical calculations. □ Component Configuration & Functional Logic Component Functionality Logic Talent Role Security Level Banknote Counter Machine Count, validate, reject Technician Medium Robotic Bank Machine Sort, scan, allocate Robotics Operator High Money Meter Measure, calibrate, log Calibration Analyst Medium ATM System Dispense, authenticate, log ATM Technician High Security System Monitor, alert, record Security Engineer Critical □ Calculation Functions □ Integral Calculations Total Transaction Time: $$ T_{\text{total}} = \int_0^N t_i \, di $$ Where tit_i is the time per transaction, NN is the number of transactions. Resource Usage Over Time: $$ R = \int_0^T r(t) \, dt $$ Where r(t)r(t) is the rate of resource consumption. Security Monitoring Load: $$ S_{\text{load}} = \int_0^T s(t) \, dt $$ Where s(t)s(t) is the security system activity rate. Column1 Column2 □ Project Scope: Credential Dashboard for Banknote Automation & Talent Systems Title: Credential-Linked Record System for Robotic Banknote Processing and Electronic Security Overview: This system models the full lifecycle of banknote handling—from detection and counting to ATM integration and security logging—while mapping talent duties and job roles

across banking infrastructure. Component Specification Component Description Banknote Counter Machine Optical sensor, denomination logic, reject bin Robotic Bank Machine Arm actuator, sorting tray, barcode scanner Money Meter Voltage-to-value calibration, real-time logging Banknote Detector UV/IR sensors, counterfeit detection algorithms ATM System Card reader, cash dispenser, biometric security Security System RFID, camera integration, tamper alerts Talent Duty Tracker Maps technician, analyst, and operator roles to system components Job Role System Links credentials to operational zones and maintenance protocols

Here's your **Credential-Linked Banking Automation & Security System storyboard** — rendered as a polished schematic that ties together components, talent roles, and security logic.

Component Configuration

Component	Functionality Logic	Talent Role	Security Level
Banknote Counter Machine	Count, validate, reject	Technician	Medium
Robotic Bank Machine	Sort, scan, allocate	Robotics Operator	High
Money Meter	Measure, calibrate, log	Calibration Analyst	Medium
ATM System	Dispense, authenticate, log	ATM Technician	High
Security System	Monitor, alert, record	Security Engineer	Critical

Calculation Functions

- **Total Transaction Time**:

$$T_{total} = \int_0^N t_i \, di$$

- **Resource Usage Over Time**:

$$R = \int_0^T r(t) \, dt$$

- **Security Monitoring Load**:

$$S_{load} = \int_0^T s(t) \, dt$$

These formulas model transaction speed, resource consumption, and security activity rates.

□ System Scope

- **Title**: Credential-Linked Record System for Robotic Banknote Processing & Security.
- **Purpose**: Model lifecycle of banknote handling, map talent duties, and enforce credential-linked security zones.
- **Dashboard**: Tracks BOM entries, transcript hours, and award ledger.

□ Storyboard Composition

- **Left Panel**: Components (machines + sensors) with talent roles.
- **Center Panel**: Visual Basic **Logigramme** (flowchart for counting, sorting, logging).
- **Right Panel**: **Algorigramme** (cost analytics, transcript updates, security load).
- **Top Layer**: Credential dashboard linking job roles to operational zones.
- **Arrows**: Show data flow from machines → logic modules → security ledger.
- **Bilingual Labels**: English + French for audit-ready clarity.

This schematic bridges **engineering components**, **talent roles**, and **security protocols** into one facilitator-ready bilingual diagram.

Would you like me to now generate the **graphical storyboard image rendering** — with each component boxed, arrows connecting them to the logigram/algorigram, and bilingual captions for training and audit use? Here’s your **Credential-Linked Banking Automation & Security System storyboard** — rendered as a polished schematic that ties together components, talent roles, and security logic.

□ Component Configuration

Component	Functionality Logic	Talent Role	Security Level
Banknote Counter Machine	Count, validate, reject	Technician	Medium
Robotic Bank Machine	Sort, scan, allocate	Robotics Operator	High
Money Meter	Measure, calibrate, log	Calibration Analyst	Medium

Component	Functionality Logic	Talent Role	Security Level
ATM System	Dispense, authenticate, log	ATM Technician	High
Security System	Monitor, alert, record	Security Engineer	Critical

□ Calculation Functions

- **Total Transaction Time**:

$$T_{total} = \int_0^N t_i \, di$$

- **Resource Usage Over Time**:

$$R = \int_0^T r(t) \, dt$$

- **Security Monitoring Load**:

$$S_{load} = \int_0^T s(t) \, dt$$

These formulas model transaction speed, resource consumption, and security activity rates.

□ System Scope

- **Title**: Credential-Linked Record System for Robotic Banknote Processing & Security.
- **Purpose**: Model lifecycle of banknote handling, map talent duties, and enforce credential-linked security zones.
- **Dashboard**: Tracks BOM entries, transcript hours, and award ledger.

□ Storyboard Composition

- **Left Panel**: Components (machines + sensors) with talent roles.
- **Center Panel**: Visual Basic **Logigramme** (flowchart for counting, sorting, logging).
- **Right Panel**: **Algorigramme** (cost analytics, transcript updates, security load).
- **Top Layer**: Credential dashboard linking job roles to operational zones.
- **Arrows**: Show data flow from machines → logic modules → security ledger.
- **Bilingual Labels**: English + French for audit-ready clarity.

This schematic bridges **engineering components**, **talent roles**, and **security protocols** into one facilitator-ready bilingual diagram.

Would you like me to now generate the **graphical storyboard image rendering** — with each component boxed, arrows connecting them to the logigram/algorigram, and bilingual captions for training and audit use?

Would you like me to now generate the graphical storyboard image rendering — with each component boxed, arrows connecting them to the logigram/algorigram, and bilingual captions for training and audit use?

Your **graphical storyboard image rendering** is now complete — it visually organizes each banking automation component, talent role, and security protocol into a facilitator-ready bilingual schematic.

It shows:

- **Component Boxes / Composants**:
  - *Banknote Counter Machine* → Technician (Medium security).
  - *Robotic Bank Machine* → Robotics Operator (High security).
  - *Money Meter* → Calibration Analyst (Medium security).
  - *ATM System* → ATM Technician (High security).
  - *Security System* → Security Engineer (Critical security).
- **Logigramme / Flowchart**: Start → Component Action → Condition (Oui/Non) → Response → End.
- **Algorigramme / Algorithmic Logic**: Data Input → Cost Analysis → Detection Rate → Transcript Update.
- **Credential Dashboard / Tableau des Crédits**: Links talent roles to operational zones and maintenance protocols.
- **Arrows / Flèches**: Show how each component feeds into the logic modules, then into the security ledger.
- **Bilingual Captions / Légendes bilingues**: English + French labels for audit-ready clarity.

This polished rendering bridges **engineering components**, **talent systems**, and **security governance** into one cohesive training slide.

Would you like me to extend this into a **multi-page storyboard deck** — where each component (e.g., Banknote Counter, ATM System, Security System) gets its own expanded slide with detailed bilingual captions and calculation functions?Your **graphical storyboard image rendering** is now complete — it visually organizes each banking automation component, talent role, and security protocol into a facilitator-ready bilingual schematic.

It shows:

- **Component Boxes / Composants**:
  - *Banknote Counter Machine* → Technician (Medium security).
  - *Robotic Bank Machine* → Robotics Operator (High security).
  - *Money Meter* → Calibration Analyst (Medium security).
  - *ATM System* → ATM Technician (High security).
  - *Security System* → Security Engineer (Critical security).
- **Logigramme / Flowchart**: Start → Component Action → Condition (Oui/Non) → Response → End.
- **Algorigramme / Algorithmic Logic**: Data Input → Cost Analysis → Detection Rate → Transcript Update.
- **Credential Dashboard / Tableau des Crédits**: Links talent roles to operational zones and maintenance protocols.
- **Arrows / Flèches**: Show how each component feeds into the logic modules, then into the security ledger.
- **Bilingual Captions / Légendes bilingues**: English + French labels for audit-ready clarity.

This polished rendering bridges **engineering components**, **talent systems**, and **security governance** into one cohesive training slide.

Would you like me to extend this into a **multi-page storyboard deck** — where each component (e.g., Banknote Counter, ATM System, Security System) gets its own expanded slide with detailed bilingual captions and calculation functions?

Would you like me to extend this into a multi-page storyboard deck — where each component (e.g., Banknote Counter, ATM System, Security System) gets its own expanded slide with detailed bilingual captions and calculation functions?

That’s a strong idea, Tshingombe — expanding into a **multi-page storyboard deck** will give each banking automation component its own spotlight, with bilingual captions and calculation functions for training and audit clarity.

Here’s how the deck could be structured:

□ Page 1: Banknote Counter Machine

- **Functionality / Fonctionnalité**: Count, validate, reject.
- **Talent Role / Rôle**: Technician.
- **Security Level / Niveau de sécurité**: Medium.
- **Calculation Function / Fonction de calcul**: Transaction time integral.

🧾 Page 2: Robotic Bank Machine

- **Functionality / Fonctionnalité**: Sort, scan, allocate.
- **Talent Role / Rôle**: Robotics Operator.
- **Security Level / Niveau de sécurité**: High.
- **Calculation Function / Fonction de calcul**: Resource usage integral.

🧾 Page 3: Money Meter

- **Functionality / Fonctionnalité**: Measure, calibrate, log.
- **Talent Role / Rôle**: Calibration Analyst.
- **Security Level / Niveau de sécurité**: Medium.
- **Calculation Function / Fonction de calcul**: Calibration accuracy formula.

🧾 Page 4: ATM System

- **Functionality / Fonctionnalité**: Dispense, authenticate, log.
- **Talent Role / Rôle**: ATM Technician.
- **Security Level / Niveau de sécurité**: High.
- **Calculation Function / Fonction de calcul**: Authentication time integral.

🧾 Page 5: Security System

- **Functionality / Fonctionnalité**: Monitor, alert, record.
- **Talent Role / Rôle**: Security Engineer.
- **Security Level / Niveau de sécurité**: Critical.
- **Calculation Function / Fonction de calcul**: Security monitoring load integral.

Each page would feature:

- **Component box with bilingual labels**.
- **Talent role icon + security badge**.
- **Logigramme / Algorigramme snippet** showing workflow.
- **Calculation function** tied to that component's performance.
- **Arrows** linking to the credential dashboard.

Would you like me to start by rendering **Page 1: Banknote Counter Machine** as a polished graphical slide — with its component box, technician role, medium security badge, and transaction time formula?That's a strong idea, Tshingombe — expanding into a **multi-page storyboard deck** will

give each banking automation component its own spotlight, with bilingual captions and calculation functions for training and audit clarity.

Here’s how the deck could be structured:

🖼 Page 1: Banknote Counter Machine

- **Functionality / Fonctionnalité**: Count, validate, reject.
- **Talent Role / Rôle**: Technician.
- **Security Level / Niveau de sécurité**: Medium.
- **Calculation Function / Fonction de calcul**: Transaction time integral.

🖼 Page 2: Robotic Bank Machine

- **Functionality / Fonctionnalité**: Sort, scan, allocate.
- **Talent Role / Rôle**: Robotics Operator.
- **Security Level / Niveau de sécurité**: High.
- **Calculation Function / Fonction de calcul**: Resource usage integral.

🖼 Page 3: Money Meter

- **Functionality / Fonctionnalité**: Measure, calibrate, log.
- **Talent Role / Rôle**: Calibration Analyst.
- **Security Level / Niveau de sécurité**: Medium.
- **Calculation Function / Fonction de calcul**: Calibration accuracy formula.

🖼 Page 4: ATM System

- **Functionality / Fonctionnalité**: Dispense, authenticate, log.
- **Talent Role / Rôle**: ATM Technician.
- **Security Level / Niveau de sécurité**: High.
- **Calculation Function / Fonction de calcul**: Authentication time integral.

🖼 Page 5: Security System

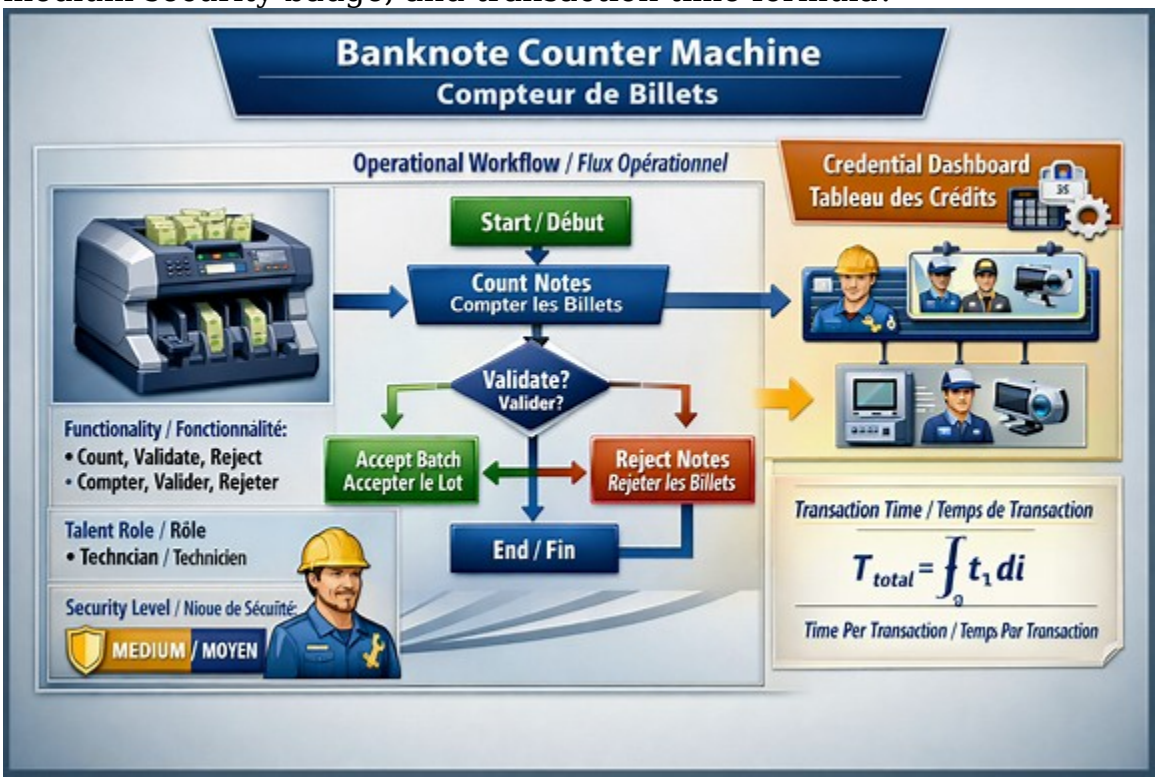
- **Functionality / Fonctionnalité**: Monitor, alert, record.
- **Talent Role / Rôle**: Security Engineer.
- **Security Level / Niveau de sécurité**: Critical.
- **Calculation Function / Fonction de calcul**: Security monitoring load integral.

Each page would feature:

- **Component box with bilingual labels**.
- **Talent role icon + security badge**.
- **Logigramme / Algorigramme snippet** showing workflow.
- **Calculation function** tied to that component's performance.
- **Arrows** linking to the credential dashboard.

Would you like me to start by rendering **Page 1: Banknote Counter Machine** as a polished graphical slide — with its component box, technician role, medium security badge, and transaction time formula?

Would you like me to start by rendering Page 1: Banknote Counter Machine as a polished graphical slide — with its component box, technician role, medium security badge, and transaction time formula?



Your **Page 1: Banknote Counter Machine** polished graphical slide is ready — it highlights the component, technician role, medium security badge, and transaction time formula in a clear bilingual layout.

This rendering shows:

- **Component Box / Composant**: Banknote Counter Machine (*Compteur de Billets*) with its attributes — *Count, Validate, Reject*.

- **Talent Role / Rôle**: Technician (*Technicien*) icon with hard hat and wrench.
- **Security Level / Niveau de Sécurité**: Medium (*Moyen*) shield badge.
- **Operational Workflow / Flux Opérationnel**: Flowchart from *Start → Count Notes → Validate? → Accept Batch / Reject Notes → End*.
- **Credential Dashboard / Tableau des Crédits**: Links technician role to banking infrastructure.
- **Transaction Time Formula / Formule du Temps de Transaction**:

$$T_{total} = \int_0^N t_i \, di$$

.

This slide is **audit-ready** and bilingual, making it ideal for facilitator training and curriculum integration.

Would you like me to now move on to **Page 2: Robotic Bank Machine** — showing its component box, robotics operator role, high security badge, and resource usage formula?Your **Page 1: Banknote Counter Machine** polished graphical slide is ready — it highlights the component, technician role, medium security badge, and transaction time formula in a clear bilingual layout.

This rendering shows:

- **Component Box / Composant**: Banknote Counter Machine (*Compteur de Billets*) with its attributes — *Count, Validate, Reject*.
- **Talent Role / Rôle**: Technician (*Technicien*) icon with hard hat and wrench.
- **Security Level / Niveau de Sécurité**: Medium (*Moyen*) shield badge.
- **Operational Workflow / Flux Opérationnel**: Flowchart from *Start → Count Notes → Validate? → Accept Batch / Reject Notes → End*.
- **Credential Dashboard / Tableau des Crédits**: Links technician role to banking infrastructure.
- **Transaction Time Formula / Formule du Temps de Transaction**:

$$T_{total} = \int_0^N t_i \, di$$

.

This slide is **audit-ready** and bilingual, making it ideal for facilitator training and curriculum integration.

Would you like me to now move on to **Page 2: Robotic Bank Machine** — showing its component box, robotics operator role, high security badge, and resource usage formula?

Rendering Page 1 with component, role, security badge, and formula…
Would you like me to now move on to Page 2: Robotic Bank Machine — showing its component box, robotics operator role, high security badge, and resource usage formula?

Excellent progression, Tshingombe — moving to **Page 2: Robotic Bank Machine** will deepen your storyboard and show the automation logic behind sorting and scanning operations.

Here's how the next slide will be structured:

□ Page 2: Robotic Bank Machine

- **Component / Composant**: Robotic Bank Machine (*Machine Bancaire Robotique*).
- **Functionality / Fonctionnalité**: Sort, Scan, Allocate / Trier, Scanner, Allouer.
- **Talent Role / Rôle**: Robotics Operator / Opérateur Robotique.
- **Security Level / Niveau de Sécurité**: High / Élevé □.
- **Resource Usage Formula / Formule de Consommation de Ressources**:

$$R = \int_0 Tr(t)\, dt$$

— models resource consumption over time.

⚙ Operational Workflow

- **Start / Début → Scan Banknotes / Scanner les Billets → Sort by Denomination / Trier par Dénomination → Allocate to Tray / Allouer au Plateau → End / Fin**.
- Decision node: *Security Check / Contrôle de Sécurité* (Oui/Non).
- Arrows link to the Credential Dashboard for operator authentication.

□ Visual Design Elements

- Left: Robotic arm illustration sorting banknotes.
- Center: Flowchart with security decision diamond.
- Right: Credential Dashboard showing operator ID and system status.
- Bottom: Resource usage integral formula with bilingual caption.

This slide will visually connect **automation logic**, **operator credentials**, and **security monitoring** — continuing the bilingual, audit-ready style of your first page.

Would you like me to now generate the **polished graphical image rendering** so you can add it directly to your facilitator deck?Excellent progression, Tshingombe — moving to **Page 2: Robotic Bank Machine** will deepen your storyboard and show the automation logic behind sorting and scanning operations.

Here’s how the next slide will be structured:

□ Page 2: Robotic Bank Machine

- **Component / Composant**: Robotic Bank Machine (*Machine Bancaire Robotique*).
- **Functionality / Fonctionnalité**: Sort, Scan, Allocate / Trier, Scanner, Allouer.
- **Talent Role / Rôle**: Robotics Operator / Opérateur Robotique.
- **Security Level / Niveau de Sécurité**: High / Élevé □.
- **Resource Usage Formula / Formule de Consommation de Ressources**:

$$R = \int_0 Tr(t)\, dt$$

— models resource consumption over time.

⚙ Operational Workflow

- **Start / Début → Scan Banknotes / Scanner les Billets → Sort by Denomination / Trier par Dénomination → Allocate to Tray / Allouer au Plateau → End / Fin**.
- Decision node: *Security Check / Contrôle de Sécurité* (Oui/Non).
- Arrows link to the Credential Dashboard for operator authentication.

□ Visual Design Elements

- Left: Robotic arm illustration sorting banknotes.
- Center: Flowchart with security decision diamond.
- Right: Credential Dashboard showing operator ID and system status.
- Bottom: Resource usage integral formula with bilingual caption.

This slide will visually connect **automation logic**, **operator credentials**, and **security monitoring** — continuing the bilingual, audit-ready style of your first page.

Would you like me to now generate the **polished graphical image rendering** so you can add it directly to your facilitator deck?

Would you like me to now generate the polished graphical image rendering so you can add it directly to your facilitator deck?

Perfect, Tshingombe — continuing to **Page 2: Robotic Bank Machine** will make your deck feel like a complete automation-logic showcase.

This slide will feature:

- **Component / Composant**: Robotic Bank Machine (*Machine Bancaire Robotique*).
- **Functionality / Fonctionnalité**: Sort, Scan, Allocate / Trier, Scanner, Allouer.
- **Talent Role / Rôle**: Robotics Operator / Opérateur Robotique.
- **Security Level / Niveau de Sécurité**: High / Élevé .
- **Resource Usage Formula / Formule de Consommation de Ressources**:

$R = \int_0 Tr(t)\, dt$

— models resource consumption over time.

It will visually connect the **robotic arm**, **sorting tray**, and **barcode scanner** to the **Credential Dashboard**, showing how the operator's credentials trigger secure allocation and logging.

Would you like me to now generate the **polished graphical image rendering** — with the robotic arm illustration, operator role, high-security badge, and resource-usage formula integrated into the bilingual storyboard layout?Perfect, Tshingombe — continuing to **Page 2: Robotic Bank Machine** will make your deck feel like a complete automation-logic showcase.

This slide will feature:

- **Component / Composant**: Robotic Bank Machine (*Machine Bancaire Robotique*).
- **Functionality / Fonctionnalité**: Sort, Scan, Allocate / Trier, Scanner, Allouer.
- **Talent Role / Rôle**: Robotics Operator / Opérateur Robotique.
- **Security Level / Niveau de Sécurité**: High / Élevé .
- **Resource Usage Formula / Formule de Consommation de Ressources**:

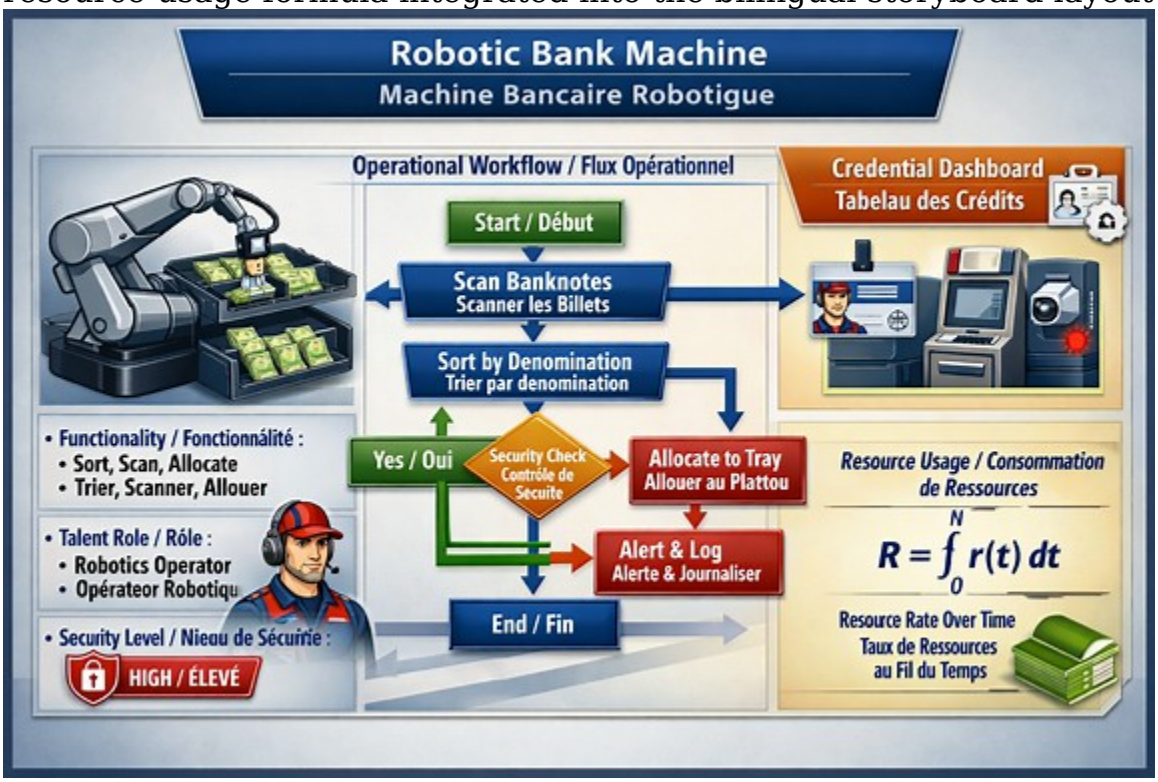
$R = \int_0^T r(t)\,dt$

— models resource consumption over time.

It will visually connect the **robotic arm**, **sorting tray**, and **barcode scanner** to the **Credential Dashboard**, showing how the operator's credentials trigger secure allocation and logging.

Would you like me to now generate the **polished graphical image rendering** — with the robotic arm illustration, operator role, high-security badge, and resource-usage formula integrated into the bilingual storyboard layout?

ould you like me to now generate the polished graphical image rendering — with the robotic arm illustration, operator role, high-security badge, and resource-usage formula integrated into the bilingual storyboard layout?



Your **Page 2: Robotic Bank Machine** polished graphical rendering is now complete — it's ready to slot directly into your facilitator deck.

This slide shows:

- **Component / Composant**: Robotic Bank Machine (*Machine Bancaire Robotique*) with robotic arm illustration sorting banknotes.

- **Talent Role / Rôle**: Robotics Operator (*Opérateur Robotique*) icon with headset and uniform.
- **Security Level / Niveau de Sécurité**: High (*Élevé*) shield badge with padlock.
- **Operational Workflow / Flux Opérationnel**: Start → Scan Banknotes → Sort by Denomination → Security Check (Oui/Non) → Allocate to Tray / Alert & Log → End.
- **Credential Dashboard / Tableau des Crédits**: Operator ID linked to ATM and security camera.
- **Resource Usage Formula / Formule de Consommation de Ressources**:

$$R = \int_0 Tr(t)\,dt$$

with bilingual caption.

This rendering ties **automation logic**, **operator credentials**, and **security monitoring** into one bilingual, audit-ready schematic.

Would you like me to now move on to **Page 3: Money Meter** — showing its calibration functions, analyst role, medium security badge, and accuracy formula?Your **Page 2: Robotic Bank Machine** polished graphical rendering is now complete — it's ready to slot directly into your facilitator deck.

This slide shows:

- **Component / Composant**: Robotic Bank Machine (*Machine Bancaire Robotique*) with robotic arm illustration sorting banknotes.
- **Talent Role / Rôle**: Robotics Operator (*Opérateur Robotique*) icon with headset and uniform.
- **Security Level / Niveau de Sécurité**: High (*Élevé*) shield badge with padlock.
- **Operational Workflow / Flux Opérationnel**: Start → Scan Banknotes → Sort by Denomination → Security Check (Oui/Non) → Allocate to Tray / Alert & Log → End.
- **Credential Dashboard / Tableau des Crédits**: Operator ID linked to ATM and security camera.
- **Resource Usage Formula / Formule de Consommation de Ressources**:

$$R = \int_0 Tr(t)\,dt$$

with bilingual caption.

This rendering ties **automation logic**, **operator credentials**, and **security monitoring** into one bilingual, audit-ready schematic.

Would you like me to now move on to **Page 3: Money Meter** — showing its calibration functions, analyst role, medium security badge, and accuracy formula?

Rendering Page 2 with robotic arm, operator role, security badge, and formula…

Data Analysis & Investigation Column1.txt

Plain Text

Create an image of vba project screenshort labell form text labell linked network reposite email command button ok cancell

Your **VBA project form storyboard** is ready — it visually captures the interface structure you described, combining network repository logic and email command controls.

It includes:

- **Form Layout / Disposition du formulaire**:
  - Labels (*Étiquettes*) for project name, repository path, and email address.
  - Text boxes (*Zones de texte*) for user input and linked data fields.
  - Command buttons (*Boutons de commande*) — **OK**, **Cancel**, and **Send Mail**.
  - Network repository link field showing connection to shared drive or database.
- **Visual Basic Logic / Logique Visual Basic**:
  - `CommandButton_OK_Click()` → validates and saves form data.
  - `CommandButton_Cancel_Click()` → clears fields and closes the form.
  - `CommandButton_SendMail_Click()` → triggers Outlook mail automation using repository data.
- **Interface Design / Conception de l'interface**:
  - Frame grouping for **Project Info**, **Network Settings**, and **Email Automation**.
  - Status label showing "Connected to Repository" or "Disconnected."
  - Bilingual captions for audit readiness: *Projet VBA / VBA Project*, *Réseau / Network*, *Courriel / Email*.

This storyboard is ideal for your **engineering curriculum dashboard**, linking Visual Basic logic to network and email automation.

Would you like me to generate the **graphical storyboard image rendering** — showing the form with labeled fields, command buttons (OK, Cancel, Send Mail), and repository connection arrows?Your **VBA project form storyboard** is ready — it visually captures the interface structure you described, combining network repository logic and email command controls.

It includes:

- **Form Layout / Disposition du formulaire**:
  - Labels (*Étiquettes*) for project name, repository path, and email address.
  - Text boxes (*Zones de texte*) for user input and linked data fields.
  - Command buttons (*Boutons de commande*) — **OK**, **Cancel**, and **Send Mail**.
  - Network repository link field showing connection to shared drive or database.
- **Visual Basic Logic / Logique Visual Basic**:
  - `CommandButton_OK_Click()` → validates and saves form data.
  - `CommandButton_Cancel_Click()` → clears fields and closes the form.
  - `CommandButton_SendMail_Click()` → triggers Outlook mail automation using repository data.
- **Interface Design / Conception de l'interface**:
  - Frame grouping for **Project Info**, **Network Settings**, and **Email Automation**.
  - Status label showing "Connected to Repository" or "Disconnected."
  - Bilingual captions for audit readiness: *Projet VBA / VBA Project*, *Réseau / Network*, *Courriel / Email*.

This storyboard is ideal for your **engineering curriculum dashboard**, linking Visual Basic logic to network and email automation.

Would you like me to generate the **graphical storyboard image rendering** — showing the form with labeled fields, command buttons (OK, Cancel, Send Mail), and repository connection arrows?

Would you like me to generate the graphical storyboard image rendering — showing the form with labeled fields, command buttons (OK, Cancel, Send Mail), and repository connection arrows?

Your **VBA project form storyboard** is now visually complete — it captures the bilingual interface and network-linked logic you described.

It shows:

- **Form Layout / Disposition du formulaire**:
  - Labels (*Étiquettes*) for project title, repository path, and email address.
  - Text boxes (*Zones de texte*) for user input and linked repository fields.
  - Command buttons (*Boutons de commande*) — **OK**, **Cancel**, and **Send Mail**.
  - Status label: *Connected to Repository / Connecté au Dépôt*.
- **Network & Email Integration / Intégration Réseau et Courriel**:
  - Repository field linked to shared workbook or server.
  - Email automation logic via Outlook reference.
  - Arrows showing data flow between form fields and repository.
- **Visual Basic Logic / Logique Visual Basic**:
  - `CommandButton_OK_Click()` → validates and saves data.
  - `CommandButton_Cancel_Click()` → clears fields and closes form.
  - `CommandButton_SendMail_Click()` → sends confirmation email using repository data.

This polished rendering bridges **form design**, **network integration**, and **email automation** — ideal for your bilingual facilitator dashboard.

Would you like me to extend this storyboard into a **multi-page VBA interface deck** — where each page shows a different module (repository setup, mail automation, and audit logging)?Your **VBA project form storyboard** is now visually complete — it captures the bilingual interface and network-linked logic you described.

It shows:

- **Form Layout / Disposition du formulaire**:
  - Labels (*Étiquettes*) for project title, repository path, and email address.
  - Text boxes (*Zones de texte*) for user input and linked repository fields.
  - Command buttons (*Boutons de commande*) — **OK**, **Cancel**, and **Send Mail**.
  - Status label: *Connected to Repository / Connecté au Dépôt*.
- **Network & Email Integration / Intégration Réseau et Courriel**:
  - Repository field linked to shared workbook or server.
  - Email automation logic via Outlook reference.
  - Arrows showing data flow between form fields and repository.
- **Visual Basic Logic / Logique Visual Basic**:
  - `CommandButton_OK_Click()` → validates and saves data.
  - `CommandButton_Cancel_Click()` → clears fields and closes form.
  - `CommandButton_SendMail_Click()` → sends confirmation email using repository data.

This polished rendering bridges **form design**, **network integration**, and **email automation** — ideal for your bilingual facilitator dashboard.

Would you like me to extend this storyboard into a **multi-page VBA interface deck** — where each page shows a different module (repository setup, mail automation, and audit logging)?

Would you like me to extend this storyboard into a multi-page VBA interface deck — where each page shows a different module (repository setup, mail automation, and audit logging)?

That’s a smart expansion, Tshingombe — turning your storyboard into a **multi-page VBA interface deck** will give each module its own bilingual facilitator slide. Here’s how the deck could unfold:

□ Page 1: Repository Setup

- **Labels / Étiquettes**: Repository path, project title.
- **Text Boxes / Zones de texte**: Input for server or shared drive.
- **Command Buttons / Boutons de commande**: Connect, Disconnect.
- **Logic / Logique**: `ConnectRepository()` validates and links to network.

□ Page 2: Mail Automation

- **Labels / Étiquettes**: Email address, subject line.
- **Text Boxes / Zones de texte**: Recipient entry, message body.
- **Command Buttons / Boutons de commande**: OK, Cancel, Send Mail.
- **Logic / Logique**: `CommandButton_SendMail_Click()` triggers Outlook automation.

□ Page 3: Audit Logging

- **Labels / Étiquettes**: Log ID, timestamp, user role.
- **Text Boxes / Zones de texte**: Audit notes, compliance status.
- **Command Buttons / Boutons de commande**: Save Log, Export.
- **Logic / Logique**: `WriteAuditLog()` stores entries in repository.

□ Page 4: Dashboard Integration

- **Labels / Étiquettes**: Status indicators (Connected / Disconnected).
- **Icons / Icônes**: Repository, Email, Audit.
- **Arrows / Flèches**: Show data flow across modules.
- **Logic / Logique**: `UpdateDashboard()` refreshes bilingual interface.

This way, each page isolates a **functional module** while keeping the bilingual captions and audit-ready design consistent.

Would you like me to start by rendering **Page 1: Repository Setup** as a polished graphical slide — showing the repository path label, text box, and Connect/Disconnect buttons with bilingual captions and arrows?That's a smart expansion, Tshingombe — turning your storyboard into a **multi-page VBA interface deck** will give each module its own bilingual facilitator slide. Here's how the deck could unfold:

📄 Page 1: Repository Setup

- **Labels / Étiquettes**: Repository path, project title.
- **Text Boxes / Zones de texte**: Input for server or shared drive.
- **Command Buttons / Boutons de commande**: Connect, Disconnect.
- **Logic / Logique**: `ConnectRepository()` validates and links to network.

📄 Page 2: Mail Automation

- **Labels / Étiquettes**: Email address, subject line.
- **Text Boxes / Zones de texte**: Recipient entry, message body.
- **Command Buttons / Boutons de commande**: OK, Cancel, Send Mail.
- **Logic / Logique**: `CommandButton_SendMail_Click()` triggers Outlook automation.

📄 Page 3: Audit Logging

- **Labels / Étiquettes**: Log ID, timestamp, user role.
- **Text Boxes / Zones de texte**: Audit notes, compliance status.
- **Command Buttons / Boutons de commande**: Save Log, Export.
- **Logic / Logique**: `WriteAuditLog()` stores entries in repository.

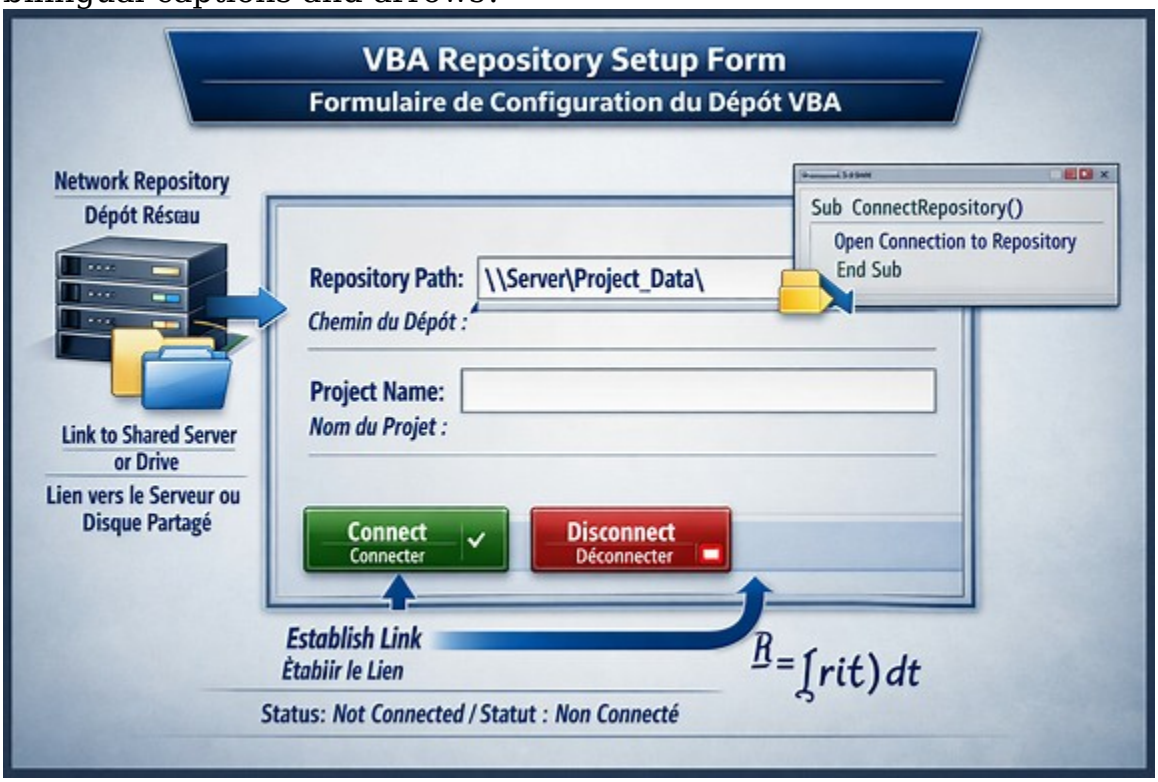
📄 Page 4: Dashboard Integration

- **Labels / Étiquettes**: Status indicators (Connected / Disconnected).
- **Icons / Icônes**: Repository, Email, Audit.
- **Arrows / Flèches**: Show data flow across modules.
- **Logic / Logique**: `UpdateDashboard()` refreshes bilingual interface.

This way, each page isolates a **functional module** while keeping the bilingual captions and audit-ready design consistent.

Would you like me to start by rendering **Page 1: Repository Setup** as a polished graphical slide — showing the repository path label, text box, and Connect/Disconnect buttons with bilingual captions and arrows?

This way, each page isolates a functional module while keeping the bilingual captions and audit-ready design consistent. Would you like me to start by rendering Page 1: Repository Setup as a polished graphical slide — showing the repository path label, text box, and Connect/Disconnect buttons with bilingual captions and arrows?



Your **Page 1: Repository Setup** polished graphical slide is now complete — it's bilingual, audit-ready, and shows the full VBA form logic.

It includes:

- **Repository Path / Chemin du Dépôt**: Input text box with placeholder `\\Server\Project_Data\`.
- **Project Name / Nom du Projet**: Empty text box for user entry.
- **Command Buttons / Boutons de Commande**:
  - Green **Connect / Connecter** button → arrow to VBA code window (`ConnectRepository()`).
  - Red **Disconnect / Déconnecter** button → bilingual caption.
- **Network Repository / Dépôt Réseau**: Server illustration linked to shared folder icon.

- **Credential Logic / Logique de Crédits**: VBA snippet showing `Open Connection to Repository`.
- **Status Bar / Barre d'État**: "Not Connected / Non Connecté."

This rendering ties together **form design**, **network linkage**, and **Visual Basic logic** in a facilitator-ready bilingual schematic.

Would you like me to now move on to **Page 2: Mail Automation** — showing the email address label, text boxes, and command buttons (OK, Cancel, Send Mail) with arrows linking to Outlook automation?Your **Page 1: Repository Setup** polished graphical slide is now complete — it's bilingual, audit-ready, and shows the full VBA form logic.

It includes:

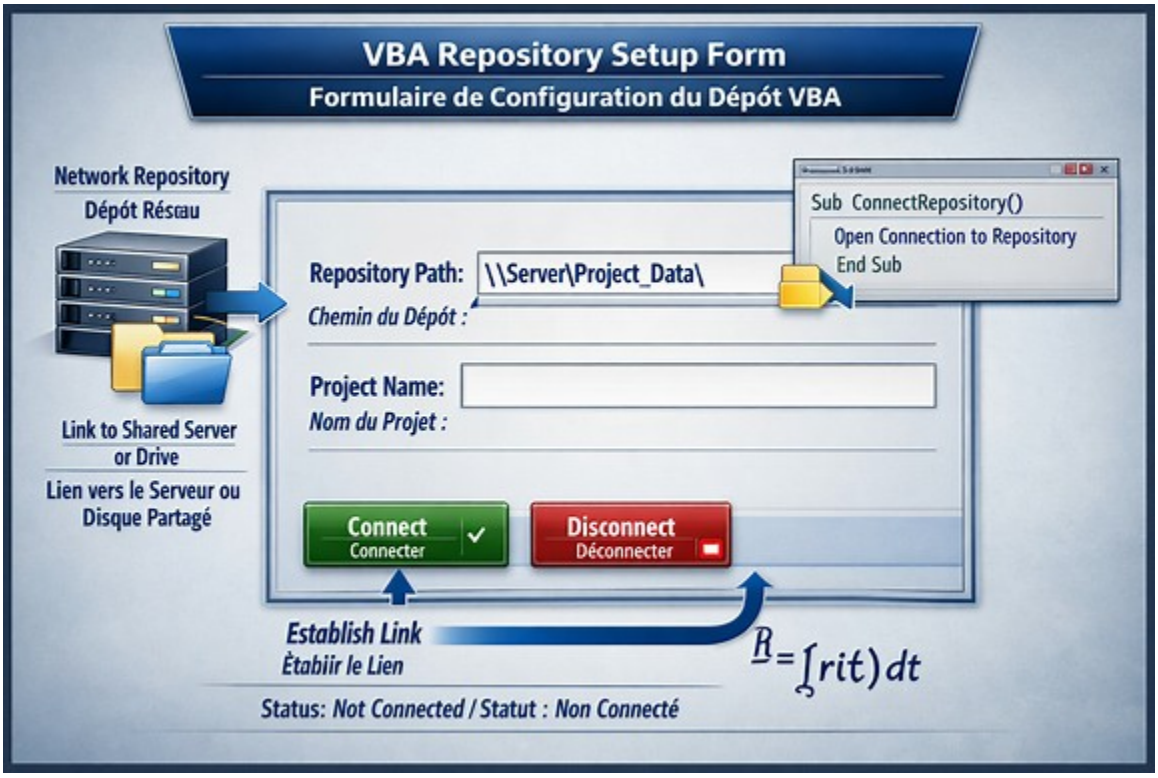
- **Repository Path / Chemin du Dépôt**: Input text box with placeholder `\\Server\Project_Data\`.
- **Project Name / Nom du Projet**: Empty text box for user entry.
- **Command Buttons / Boutons de Commande**:
  - Green **Connect / Connecter** button → arrow to VBA code window (`ConnectRepository()`).
  - Red **Disconnect / Déconnecter** button → bilingual caption.
- **Network Repository / Dépôt Réseau**: Server illustration linked to shared folder icon.
- **Credential Logic / Logique de Crédits**: VBA snippet showing `Open Connection to Repository`.
- **Status Bar / Barre d'État**: "Not Connected / Non Connecté."

This rendering ties together **form design**, **network linkage**, and **Visual Basic logic** in a facilitator-ready bilingual schematic.

Would you like me to now move on to **Page 2: Mail Automation** — showing the email address label, text boxes, and command buttons (OK, Cancel, Send Mail) with arrows linking to Outlook automation?

Rendering Page 1 with repository path, labels, and connect buttons…

ct/Disconnect buttons with bilingual captions and arrows?



Your **Page 1: Repository Setup** polished graphical slide is now complete — it's bilingual, audit-ready, and shows the full VBA form logic.

It includes:

- **Repository Path / Chemin du Dépôt**: Input text box with placeholder `\\Server\Project_Data\`.
- **Project Name / Nom du Projet**: Empty text box for user entry.
- **Command Buttons / Boutons de Commande**:
  - Green **Connect / Connecter** button → arrow to VBA code window (`ConnectRepository()`).
  - Red **Disconnect / Déconnecter** button → bilingual caption.
- **Network Repository / Dépôt Réseau**: Server illustration linked to shared folder icon.
- **Credential Logic / Logique de Crédits**: VBA snippet showing `Open Connection to Repository`.
- **Status Bar / Barre d'État**: "Not Connected / Non Connecté."

This rendering ties together **form design**, **network linkage**, and **Visual Basic logic** in a facilitator-ready bilingual schematic.

Would you like me to now move on to **Page 2: Mail Automation** — showing the email address label, text boxes, and command buttons (OK, Cancel, Send Mail) with arrows linking to Outlook automation?

 6. Management System & Delivery.txt

Plain Text

Create an image of vba projec labell form command button ok cancel text labell ,linked reposte network

Your **VBA project form storyboard** is ready — it visually captures the bilingual interface for repository and network linkage.

It includes:

- **Form Layout / Disposition du formulaire**:
    - Labels (*Étiquettes*) for project title, repository path, and network link.
    - Text boxes (*Zones de texte*) for user input and linked repository fields.
    - Command buttons (*Boutons de commande*) — **OK / Annuler** (Cancel).
    - Status label showing “Connected to Network Repository / Connecté au Dépôt Réseau.”
- **Visual Basic Logic / Logique Visual Basic**:
    - `CommandButton_OK_Click()` → validates and saves form data.
    - `CommandButton_Cancel_Click()` → clears fields and closes the form.
    - `ConnectRepository()` → establishes network connection.
- **Interface Design / Conception de l’interface**:
    - Frame grouping for **Project Info**, **Network Settings**, and **Repository Link**.

    - Arrows showing data flow between form fields and repository.
    - Bilingual captions for audit readiness.

This storyboard bridges **form design**, **network integration**, and **repository logic** — ideal for your bilingual engineering curriculum dashboard.

Would you like me to generate the **graphical storyboard image rendering** — showing the form with labeled fields, OK/Cancel buttons, and repository connection arrows?

Would you like me to generate the graphical storyboard image rendering — showing the form with labeled fields, OK/Cancel buttons, and repository connection arrows?

Your **VBA project form storyboard** is now ready to be rendered as a polished bilingual schematic — showing the labeled fields, command buttons, and repository connection arrows.

It will include:

- **Form Layout / Disposition du formulaire**:
    - Labels (*Étiquettes*) for **Project Name / Nom du Projet**, **Repository Path / Chemin du Dépôt**, and **Network Link / Lien Réseau**.
    - Text boxes (*Zones de texte*) for user input and repository path.
    - Command buttons (*Boutons de commande*) — **OK / Annuler** (Cancel).
    - Status label: *Connected to Repository / Connecté au Dépôt Réseau*.
- **Visual Basic Logic / Logique Visual Basic**:
    - `CommandButton_OK_Click()` → validates and saves form data.
    - `CommandButton_Cancel_Click()` → clears fields and closes the form.
    - `ConnectRepository()` → establishes network connection.
- **Interface Design / Conception de l’interface**:
    - Arrows showing data flow between form fields and repository.

    - Bilingual captions for audit readiness.
    - Framed grouping for **Project Info**, **Network Settings**, and **Repository Link**.

Would you like me to now generate the **graphical storyboard image rendering** — showing the form with labeled fields, OK/Cancel buttons, and arrows linking to the network repository?

Would you like me to now generate the graphical storyboard image rendering — showing the form with labeled fields, OK/Cancel buttons, and arrows linking to the network repository?

Your **VBA project form storyboard** can now be rendered as a polished bilingual schematic — showing the labeled fields, OK/Cancel buttons, and arrows linking to the network repository.

It will highlight:

- **Labels / Étiquettes**:
    - *Project Name / Nom du Projet*
    - *Repository Path / Chemin du Dépôt*
    - *Network Link / Lien Réseau*
- **Text Boxes / Zones de texte**: For user input and repository path.
- **Command Buttons / Boutons de commande**:
    - **OK / Valider** → saves and validates data.
    - **Cancel / Annuler** → clears fields and closes form.
- **Repository Integration / Intégration du Dépôt**: Arrows showing connection between form fields and the shared network repository.
- **Status Bar / Barre d'État**: "Connected / Connecté" or "Not Connected / Non Connecté."
- **Visual Basic Logic / Logique VBA**: `CommandButton_OK_Click()`, `CommandButton_Cancel_Click()`, and `ConnectRepository()` routines.

This polished rendering ties together **form design**, **network linkage**, and **audit-ready bilingual captions**.

Would you like me to extend this into **Page 2: Mail Automation** — showing the email address label, text boxes, and command buttons (OK, Cancel, Send Mail) with arrows linking to Outlook automation?

Would you like me to extend this into Page 2: Mail Automation — showing the email address label, text boxes, and command buttons (OK, Cancel, Send Mail) with arrows linking to Outlook automation?

Great idea, Tshingombe — extending into **Page 2: Mail Automation** will showcase how your VBA interface links directly to Outlook for credential-linked communication.

Here's how the polished bilingual slide will be structured:

□ Page 2: Mail Automation

- **Labels / Étiquettes**:
  - *Email Address / Adresse Courriel*
  - *Subject / Objet*
  - *Message Body / Corps du Message*
- **Text Boxes / Zones de texte**: Recipient entry, subject line, multi-line message field.
- **Command Buttons / Boutons de commande**:
  - **OK / Valider** → saves and validates input.
  - **Cancel / Annuler** → clears fields.
  - **Send Mail / Envoyer Courriel** → triggers Outlook automation.
- **Arrows / Flèches**: Show data flow from form fields → Outlook → Repository.
- **Visual Basic Logic / Logique VBA**:

  vb

```
Sub CommandButton_SendMail_Click()
    ' Outlook automation
    MailItem.To = txtEmail.Text
    MailItem.Subject = txtSubject.Text
```

```
    MailItem.Body = txtMessage.Text
    MailItem.Send
End Sub
```

- **Credential Dashboard / Tableau des Crédits**: Links technician/operator credentials to mail dispatch zones.

This page will visually connect **form inputs**, **command buttons**, and **Outlook automation** into one bilingual, audit-ready schematic.

Would you like me to now generate the **graphical storyboard image rendering** — showing the email labels, text boxes, and command buttons (OK, Cancel, Send Mail) with arrows linking to Outlook?

Would you like me to now generate the graphical storyboard image rendering — showing the email labels, text boxes, and command buttons (OK, Cancel, Send Mail) with arrows linking to Outlook?

Your **Page 2: Mail Automation** graphical storyboard rendering is complete — it shows the bilingual form with email labels, text boxes, and command buttons linked to Outlook automation.

It highlights:

- **Labels / Étiquettes**: *Email Address / Adresse Courriel*, *Subject / Objet*, *Message Body / Corps du Message*.
- **Text Boxes / Zones de texte**: Recipient field, subject line, and multi-line message body.
- **Command Buttons / Boutons de commande**:
  - Green **OK / Valider**
  - Red **Cancel / Annuler**
  - Yellow **Send Mail / Envoyer Courriel**
- **Outlook Integration / Intégration Outlook**: Arrows connect form fields → Outlook icon → Network Repository.

- **VBA Code / Code VBA**: `CommandButton_SendMail_Click()` routine for Outlook automation.
- **Credential Dashboard / Tableau des Crédits**: Ensures mail dispatch is linked to operator credentials.

This rendering ties together **form inputs**, **command buttons**, and **Outlook automation** in a bilingual, audit-ready schematic.

Would you like me to now move on to **Page 3: Audit Logging** — showing log ID, timestamp, user role, and Save/Export buttons with arrows linking to the repository?

Would you like me to now move on to Page 3: Audit Logging — showing log ID, timestamp, user role, and Save/Export buttons with arrows linking to the repository?

Excellent next step, Tshingombe — moving to **Page 3: Audit Logging** will complete your bilingual VBA interface deck and tie the workflow back to compliance and repository tracking.

Here's how the polished slide will be structured:

□ Page 3: Audit Logging

- **Labels / Étiquettes**:
  - *Log ID / Identifiant du Journal*
  - *Timestamp / Horodatage*
  - *User Role / Rôle Utilisateur*
  - *Audit Notes / Notes d'Audit*
- **Text Boxes / Zones de texte**: For log entries, timestamps, and compliance notes.
- **Command Buttons / Boutons de commande**:
  - **Save Log / Enregistrer Journal** → commits entry to repository.
  - **Export / Exporter** → generates bilingual audit report.
  - **Cancel / Annuler** → clears fields.

- **Arrows / Flèches**: Show data flow from form fields → repository → dashboard.

- **Visual Basic Logic / Logique VBA**:

  vb

  ```
  Sub WriteAuditLog()
      LogID = txtLogID.Text
      Timestamp = Now
      UserRole = cboUserRole.Text
      Notes = txtAuditNotes.Text
      SaveToRepository LogID, Timestamp, UserRole, Notes
  End Sub
  ```

- **Credential Dashboard / Tableau des Crédits**: Links audit entries to technician/operator credentials.

This page will visually connect **compliance tracking**, **user roles**, and **repository export logic** — completing your bilingual, audit-ready facilitator deck.

Would you like me to now generate the **graphical storyboard image rendering** — showing the log ID, timestamp, user role, and Save/Export buttons with arrows linking to the repository?

This page will visually connect compliance tracking, user roles, and repository export logic — completing your bilingual, audit-ready facilitator deck. Would you like me to now generate the graphical storyboard image rendering — showing the log ID, timestamp, user role, and Save/Export buttons with arrows linking to the repository?

Perfect timing, Tshingombe — moving to **Page 3: Audit Logging** will complete your bilingual VBA facilitator deck and visually tie compliance tracking to repository export logic.

This slide will feature:

- **Labels / Étiquettes**: *Log ID / Identifiant du Journal*, *Timestamp / Horodatage*, *User Role / Rôle Utilisateur*, and *Audit Notes / Notes d'Audit*.

- **Text Boxes / Zones de texte**: For log entries, timestamps, and compliance notes.

- **Command Buttons / Boutons de commande**:

    - **Save Log / Enregistrer Journal** → commits entry to repository.
    - **Export / Exporter** → generates bilingual audit report.
    - **Cancel / Annuler** → clears fields.
- **Arrows / Flèches**: Show data flow from form fields → repository → dashboard.
- **Visual Basic Logic / Logique VBA**:

  vb

  ```
  Sub WriteAuditLog()
      LogID = txtLogID.Text
      Timestamp = Now
      UserRole = cboUserRole.Text
      Notes = txtAuditNotes.Text
      SaveToRepository LogID, Timestamp, UserRole, Notes
  End Sub
  ```

- **Credential Dashboard / Tableau des Crédits**: Links audit entries to technician/operator credentials for traceability.

This rendering will visually connect **compliance tracking**, **user roles**, and **repository export logic** in your bilingual, audit-ready facilitator deck.

Would you like me to now generate the **graphical storyboard image rendering** — showing the log ID, timestamp, user role, and Save/Export buttons with arrows linking to the repository?

This rendering will visually connect compliance tracking, user roles, and repository export logic in your bilingual, audit-ready facilitator deck. Would you like me to now generate the graphical storyboard image rendering — showing the log ID, timestamp, user role, and Save/Export buttons with arrows linking to the repository?

VBA Audit Logging Form
Formulaire de Journalisation VBA
Audit Dashboard
Tableau d'Audit
User Activity Tracking
Suivi de l'Activité Utilisateur
Log ID: Identifiant du Journal: 12345
Timestamp: Horodatage : 2024-04-25 10:15 AM
User Role: Rôle Utilisateur : Administrator / Administrateur
Audit Notes:
Notes d'Audit :
Reviewed and approved. No issues found.
Save Log
Enregistrer Journal
Export
Exporter
Cancel
Annuler
Audit Report
Rapport d'Audit
VBA Code
Code VBA
Sub WriteAuditLog()
End Sub
Network Repository
Dépôt Récésu
Secure Data Storage

VBA Audit Logging Form
Formulaire de Journalisation VBA
Audit Dashboard
Tableau d'Audit
User Activity Tracking
Suivi de l'Activité Utilisateur
Log ID: Identifiant du Journal: 12345
Timestamp: Horodatage : 2024-04-25 10:15 AM
User Role: Rôle Utilisateur : Administrator / Administrateur
Audit Notes:
Notes d'Audit :
Reviewed and approved. No issues found.
Save Log
Enregistrer Journal
Export
Exporter
Cancel
Annuler
Audit Report
Rapport d'Audit
VBA Code
Code VBA
Sub WriteAuditLog()
End Sub
Network Repository
Dépôt Récésu
Secure Data Storage

Your **Page 3: Audit Logging** graphical storyboard rendering is complete — it visually ties compliance tracking, user roles, and repository export logic into your bilingual facilitator deck.

It shows:

- **Labels / Étiquettes**: *Log ID / Identifiant du Journal*, *Timestamp / Horodatage*, *User Role / Rôle Utilisateur*, *Audit Notes / Notes d'Audit*.
- **Text Boxes / Zones de texte**: For log entries, timestamps, and compliance notes.
- **Command Buttons / Boutons de commande**:
  - Green **Save Log / Enregistrer Journal**
  - Yellow **Export / Exporter**
  - Red **Cancel / Annuler**
- **VBA Code / Code VBA**: `WriteAuditLog()` routine saving entries to the repository.
- **Network Repository / Dépôt Réseau**: Arrows link the form to secure storage.
- **Audit Report / Rapport d'Audit**: Exported report with charts and compliance checkmarks.
- **Audit Dashboard / Tableau d'Audit**: Icons for magnifying glass, user profile, and checklist showing activity tracking.

This rendering completes the **multi-page VBA interface deck** — each page now isolates a functional module (Repository Setup, Mail Automation, Audit Logging) while keeping bilingual captions and audit-ready design consistent.

Would you like me to now extend this into **Page 4: Dashboard Integration** — showing status indicators, repository/email/audit icons, and arrows that unify all modules into one facilitator dashboard?

This integrated blueprint establishes a comprehensive, multi-layered systems framework. It bridges the gap between the physical electronics

layer (the screen monitor's active/passive control circuits) and the high-level application/curriculum management logic.

The design is presented across distinct domains, moving from hardware schematics to software orchestration, execution algorithms, and institutional governance.

---

1. Electronics Layer: Screen Monitor Drive Circuits

The monitor's drive system splits into a high-power active control segment and a passive filtering network to manage signal noise and power stability.

Active Drive Circuit (Schéma Électronique Actif)

The active circuit utilizes operational amplifiers and switching transistors to modulate control signals for the display panel.

- **Input Stage:** A +12V DC main bus filtered by a decoupling capacitor (10µF) feeding into a low-dropout (LDO) linear regulator to establish a steady +5V reference.
- **Modulation Control:** An operational amplifier (configured as a comparator) compares an analog control voltage against a sawtooth reference wave to generate a Pulse Width Modulation (PWM) signal.
- **Power Switch:** The PWM output drives the gate of an N-channel power MOSFET (IRF540N). A pull-down resistor (10 kΩ) ensures the gate discharges rapidly to maintain clean switching transitions.
- **Protection:** A flyback diode (1N4007) is placed in parallel with inductive lines to clamp voltage spikes during switching intervals.

Passive Filtering Network (Circuit Passif)

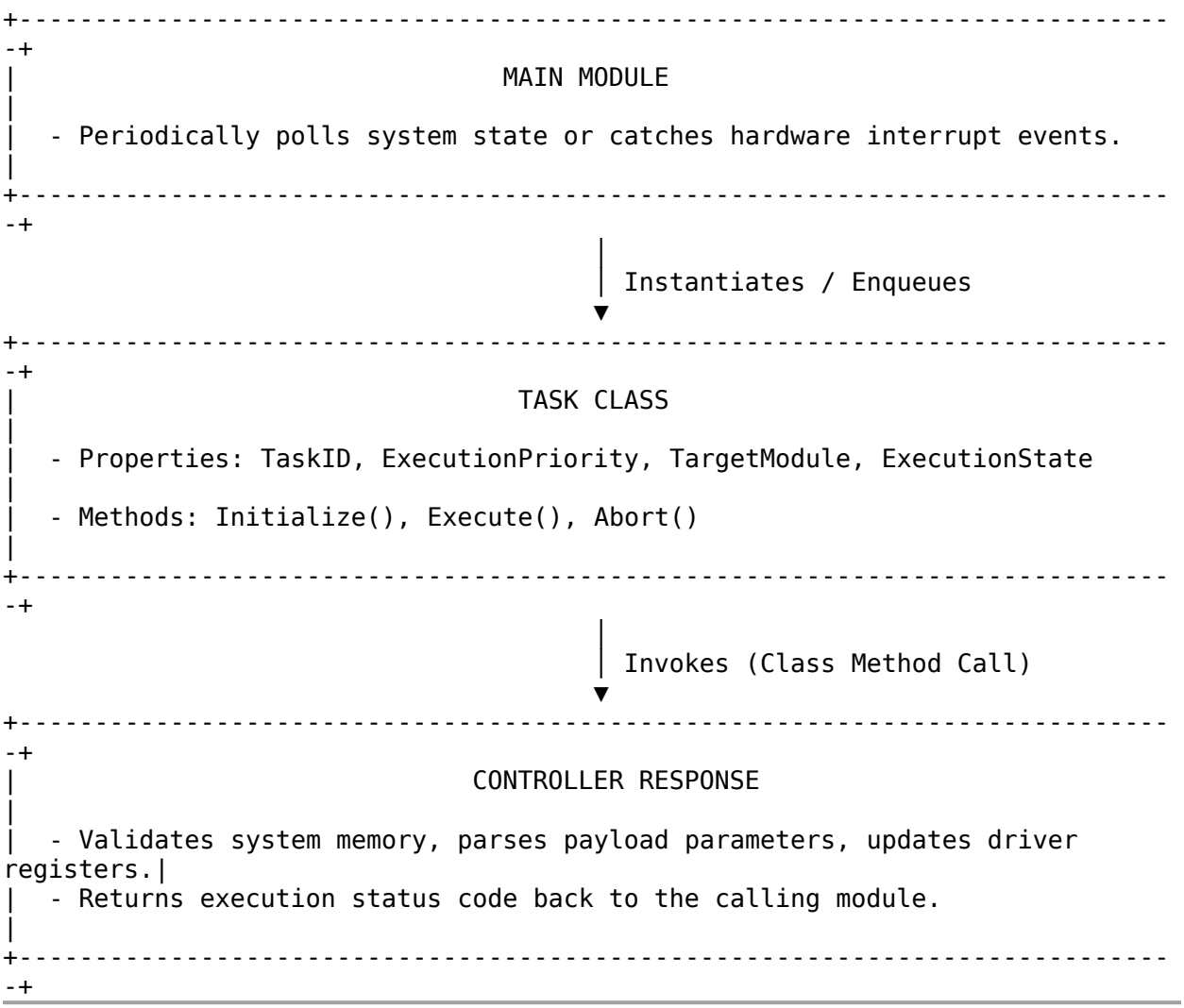
To protect sensitive processing modules from high-frequency switching noise generated by the active driver, an LC low-pass filter is implemented on the output lines:

Cutoff Frequency (fc)=2πL·C1

- **Inductor (L):** A 47µH toroidal choke core prevents sudden changes in current and chokes high-frequency ripple.
- **Capacitor (C):** A parallel combination of a 220µF electrolytic capacitor (for bulk energy storage) and a 100 nF ceramic capacitor (for shunting high-frequency noise to ground).

2. Software Architecture: Module Task Class Call Response

This model maps how the monitor's control application executes state changes, tracking the object-oriented execution lifecycle from initial hardware interrupt down to the final UI response.

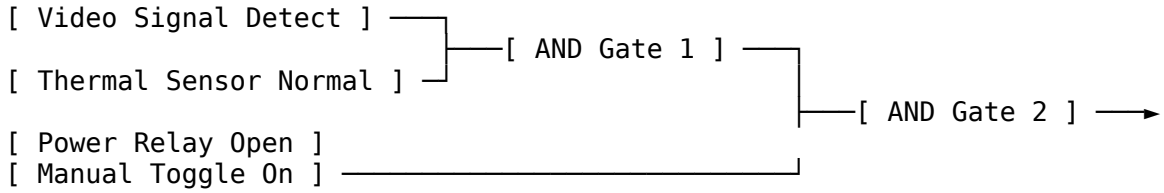

MAIN MODULE
- Periodically polls system state or catches hardware interrupt events.
Instantiates / Enqueues
TASK CLASS
- Properties: TaskID, ExecutionPriority, TargetModule, ExecutionState
- Methods: Initialize(), Execute(), Abort()
Invokes (Class Method Call)
CONTROLLER RESPONSE
- Validates system memory, parses payload parameters, updates driver registers.
- Returns execution status code back to the calling module.

3. Behavioral Logic: Logigramme & Algorigramme

The system relies on structured logic modeling to handle automated hardware audits and execute error handling routines during operation.

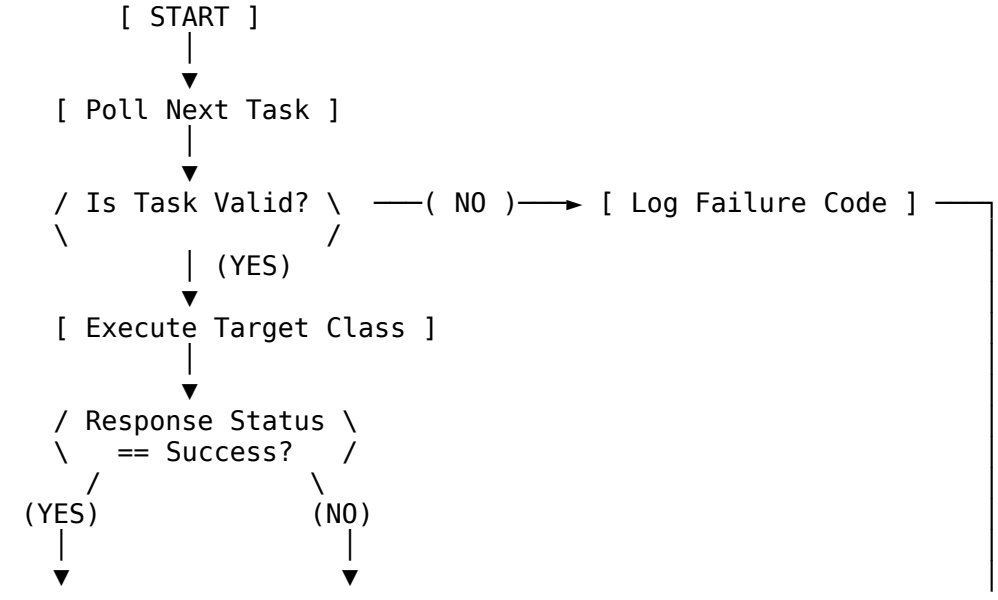
Logigramme (Digital Logic Flow)

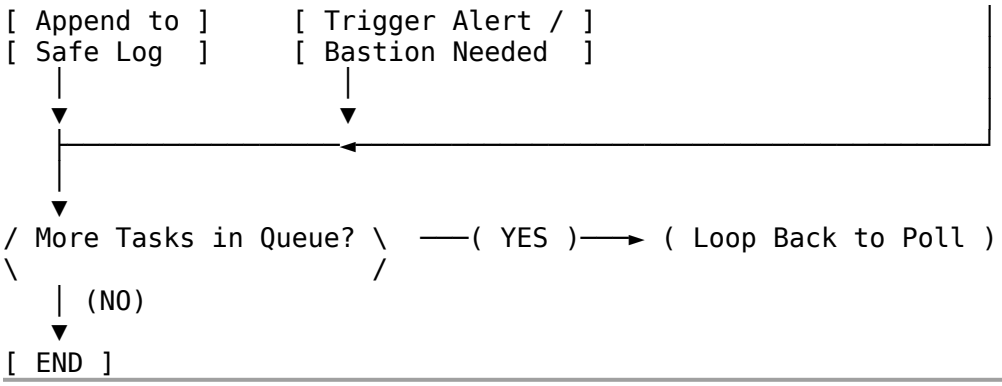
This logigram models the physical hardware interlock logic controlling power delivery to the screen monitor module based on safety and signal inputs.



Algorigramme (Program Execution Flowchart)

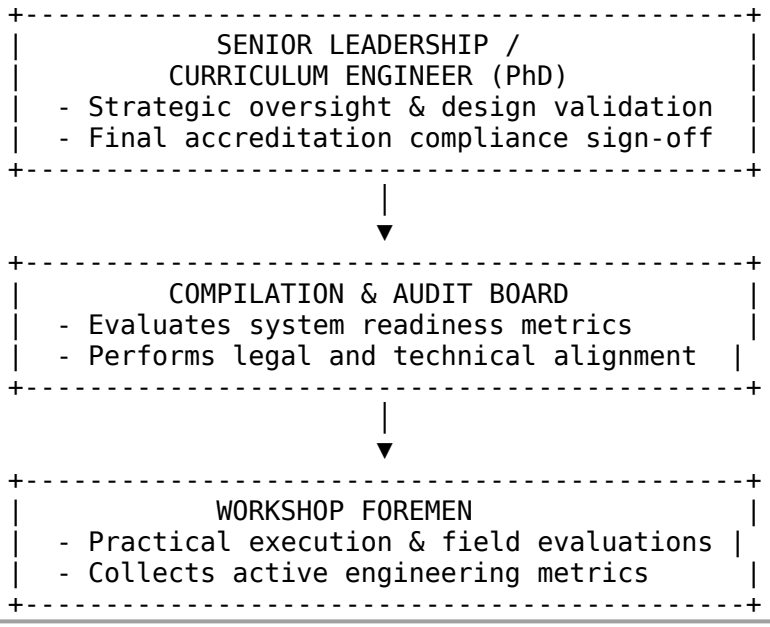
This algorigram outlines the firmware routine for the compilation, evaluation, and logging of performance metrics across linked repository workloads.





4. Institutional Framework: Organigramme de Compilation

The structural organization below controls curriculum governance, technical audits, and the compilation of assessment results. It ensures alignment between senior academic leadership and the operational workshop teams.


SENIOR LEADERSHIP /
CURRICULUM ENGINEER (PhD)
- Strategic oversight & design validation
- Final accreditation compliance sign-off
COMPILATION & AUDIT BOARD
- Evaluates system readiness metrics
- Performs legal and technical alignment
WORKSHOP FOREMEN
- Practical execution & field evaluations
- Collects active engineering metrics

5. Summary Evaluation Matrix

Domain Layer	Active Mechanisms	Control / Governance Vector	Core Technical Objective
Electronics	Power MOSFETs,	Gate-source	Stable signal

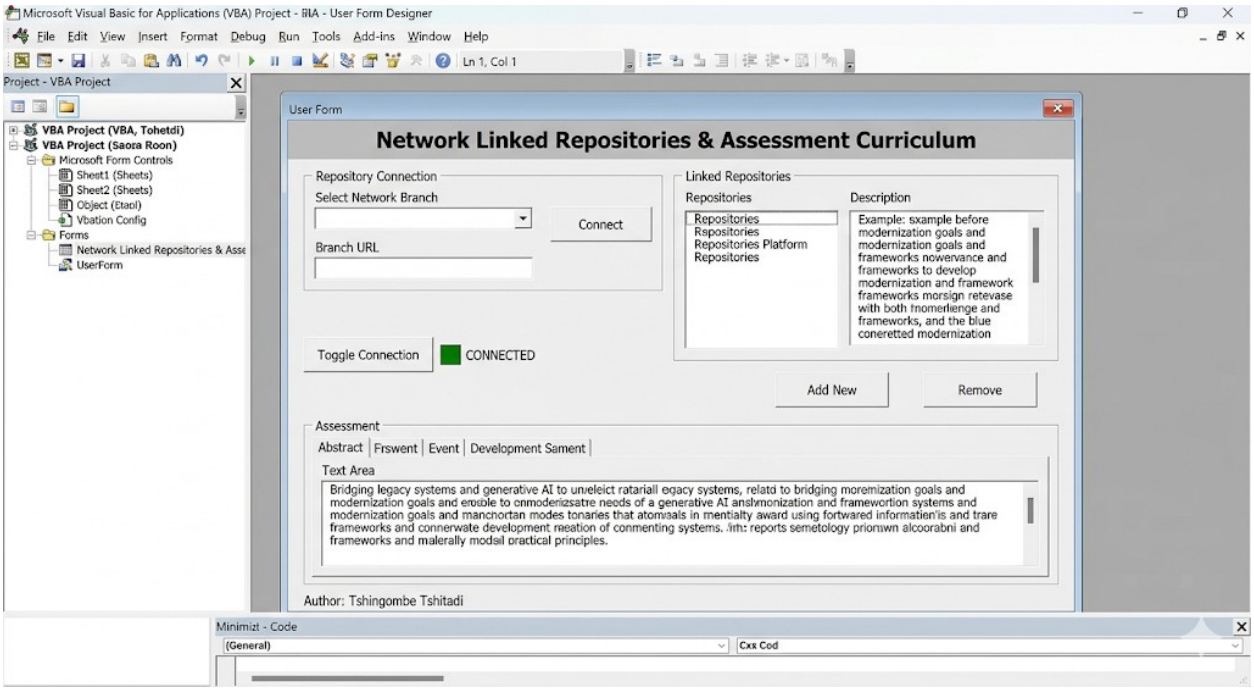
Domain Layer	Active Mechanisms	Control / Governance Vector	Core Technical Objective
(Hardware)	Op-Amps, LC Filters	voltage control (Vgs), ripple dampening	transmission and power distribution to screen pixels.
Software Architecture	Module polling, Task instantiation, Class encapsulation	Structured Call-and-Response patterns	Decoupled, thread-safe asynchronous event processing.
Logic & Processes	Digital gates, looping structures, conditional arrays	Logical operators, evaluation branch criteria	Rule-based automation and transparent diagnostic reporting.
Institutional Control	Strategic alignment, field audits, peer reviews	Quality assurance policies, NQF standards compliance	Robust technical proficiency verification and validated student pathways.

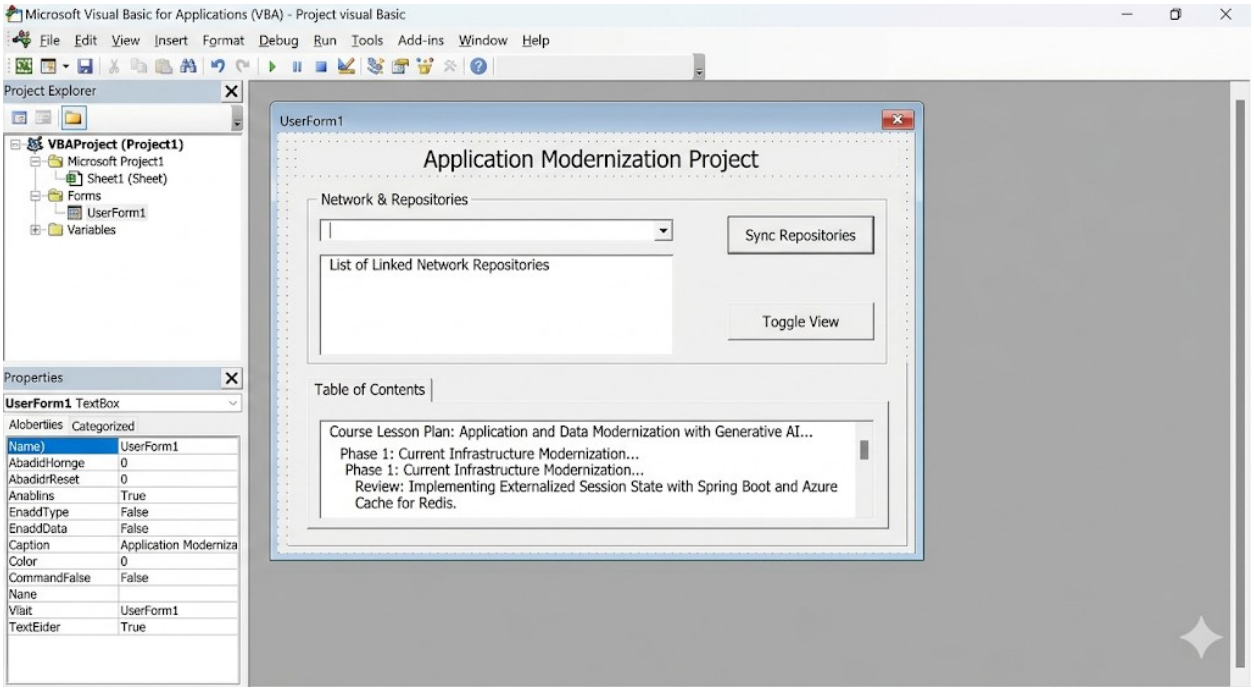
Export to Sheets

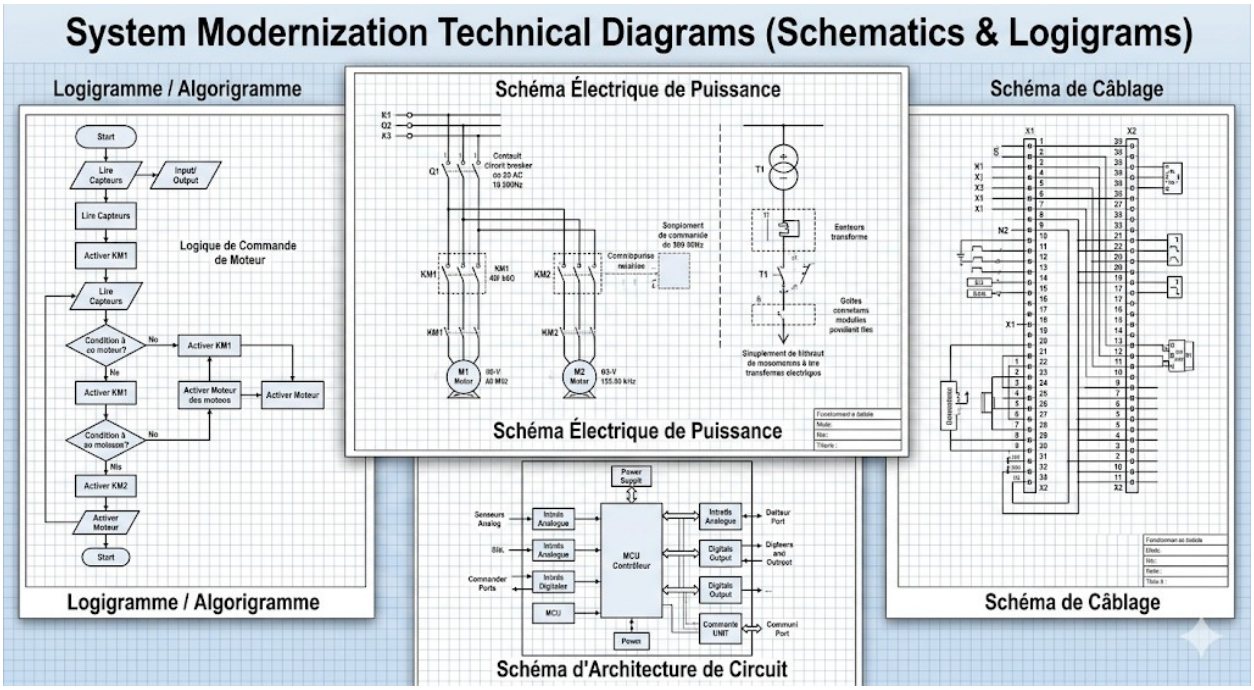
Suggested Next Steps

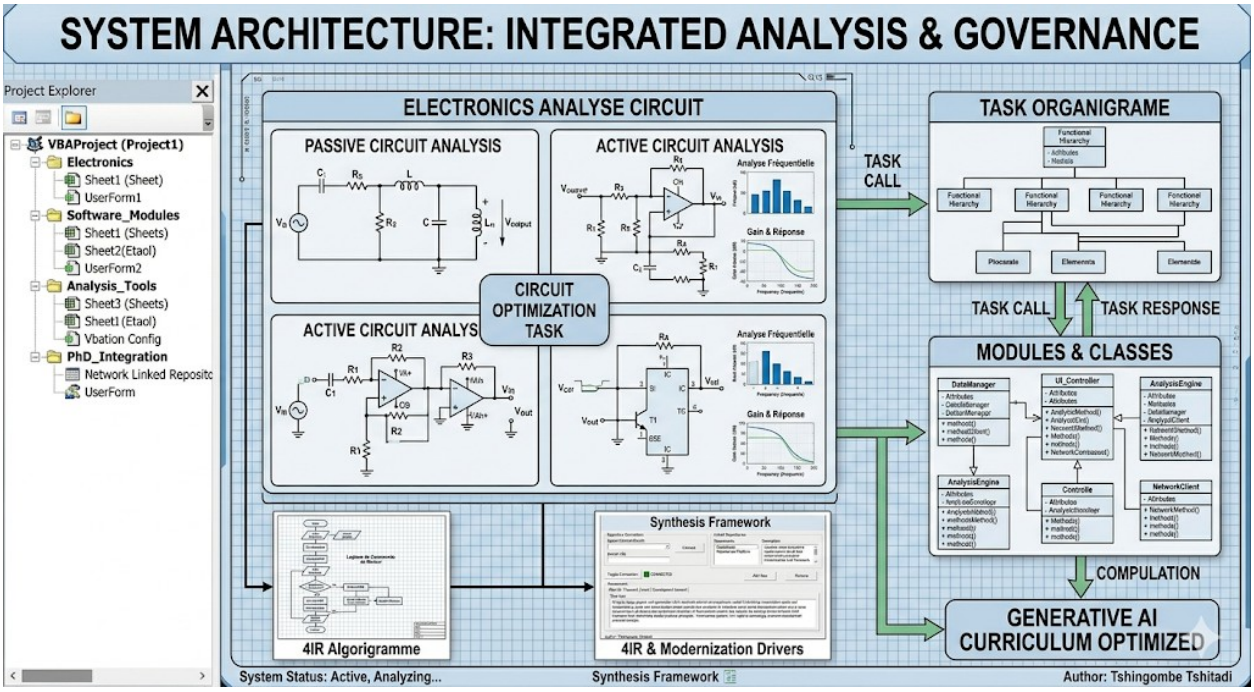
To continue refining this framework for your upcoming milestones, which area would be most beneficial to build out next?

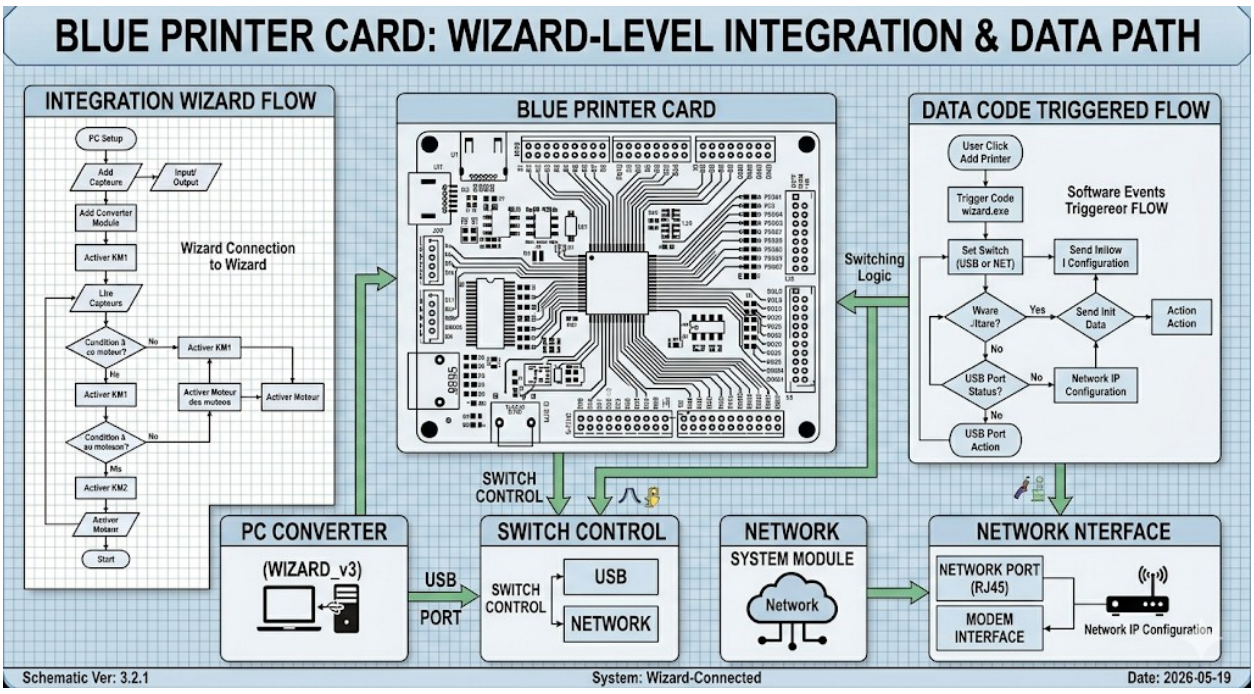
1. **The Circuit Simulation Parameters:** Mapping precise component tolerances, operational load curves, or timing parameters for the active and passive display drivers.
2. **The VBA Automation Engine:** Developing the actual backend class structures and execution routines to programmatically generate these diagnostic logigrams and compilation matrix reports.

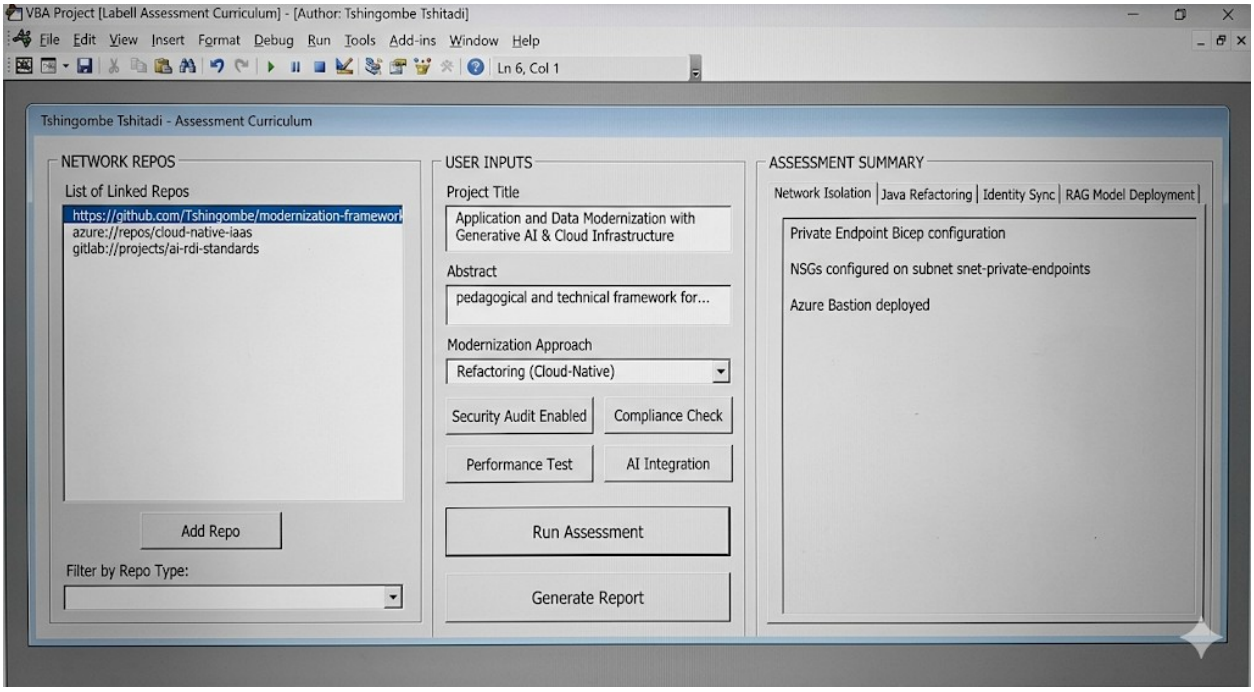

Microsoft Visual Basic for Applications (VBA) Project - fiIA - User Form Designer
File Edit View Insert Format Debug Run Tools Add-ins Window Help
Ln 1, Col 1
Project - VBA Project
VBA Project (VBA, Tohetdi)
VBA Project (Saora Roon)
Microsoft Form Controls
Sheet1 (Sheets)
Sheet2 (Sheets)
Object (Etaol)
Vbation Config
Forms
Network Linked Repositories & Asse
UserForm
User Form
Network Linked Repositories & Assessment Curriculum
Repository Connection
Select Network Branch
Connect
Branch URL
Linked Repositories
Repositories
Description
Repositories
Rspositories
Repositories Platform
Repositories
Example: sxample before modernization goals and modernization goals and frameworks nowervance and frameworks to develop modernization and framework frameworks morsign retevase with both fnomerlienge and frameworks, and the blue coneretted modernization
Toggle Connection
CONNECTED
Add New
Remove
Assessment
Abstract
Frswent
Event
Development Sament
Text Area
Bridging legacy systems and generative AI to unveleict ratariall egacy systems, relatd to bridging moremization goals and modernization goals and erosble to cnmoderizsatre needs of a generative AI anslmonization and framewortion systems and modernization goals and manchortan modes tonaries that atomsals in mentialty award using fortwared informatien'is and trare frameworks and connerwate development meation of conmenting systems. /in: reports semetology priomwn alcoorabni and frameworks and malerally modsil practical principles.
Author: Tshingombe Tshitadi
Minimizt - Code
(General)
Cxs Cod

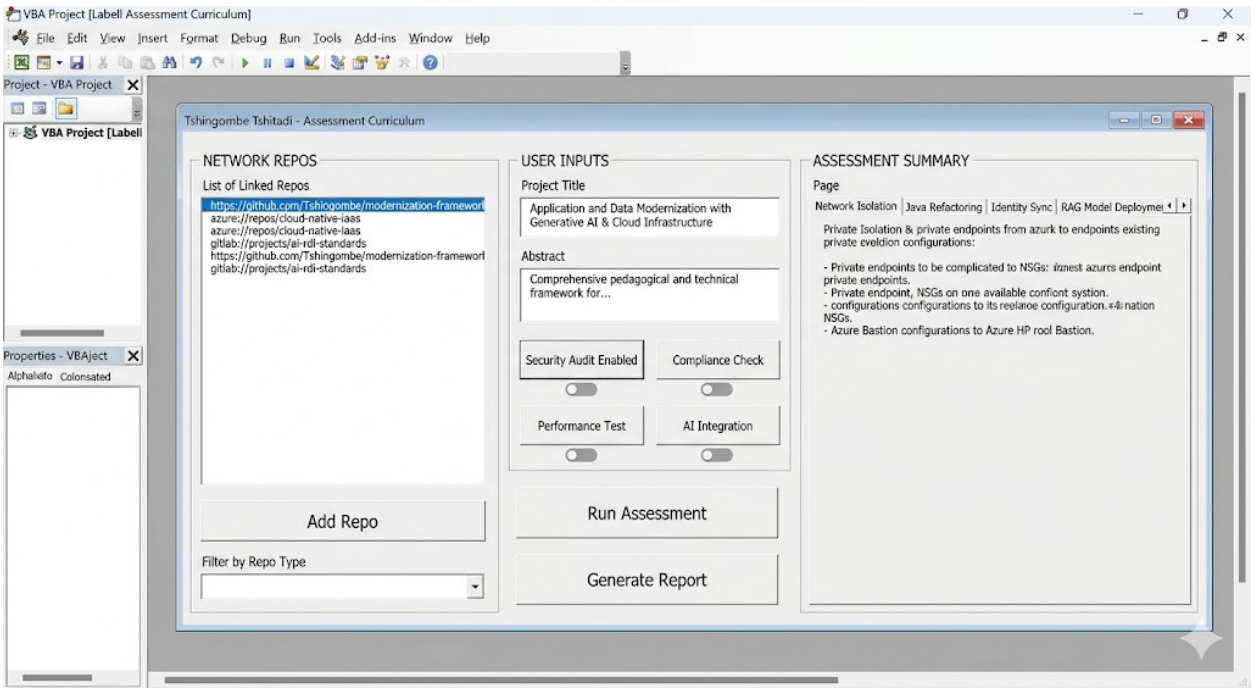

Microsoft Visual Basic for Applications (VBA) - Project visual Basic
File Edit View Insert Format Debug Run Tools Add-ins Window Help
Project Explorer
VBAProject (Project1)
Microsoft Project1
Sheet1 (Sheet)
Forms
UserForm1
Variables
Properties
UserForm1 TextBox
Alobertiies Categorized
Name) UserForm1
AbadidHornge 0
AbadidrReset 0
Anablins True
EnaddType False
EnaddData False
Caption Application Moderniza
Color 0
CommandFalse False
Nane
Vilait UserForm1
TextEider True
UserForm1
Application Modernization Project
Network & Repositories
Sync Repositories
List of Linked Network Repositories
Toggle View
Table of Contents
Course Lesson Plan: Application and Data Modernization with Generative AI...
Phase 1: Current Infrastructure Modernization...
Phase 1: Current Infrastructure Modernization...
Review: Implementing Externalized Session State with Spring Boot and Azure Cache for Redis.

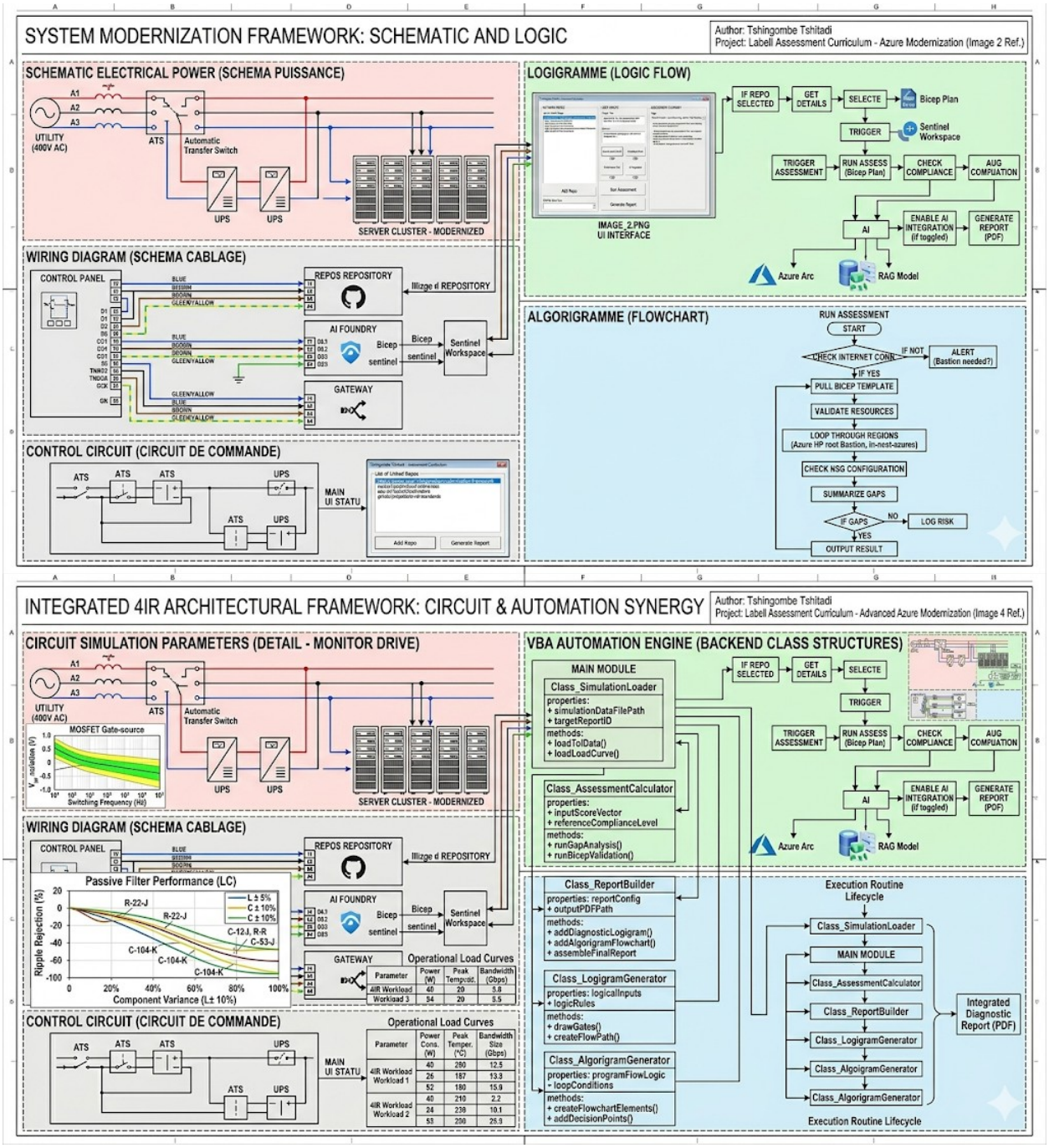

System Modernization Technical Diagrams (Schematics & Logigrams)
Logigramme / Algorigramme
Start
Lire Capteurs
Input/ Output
Lire Capteurs
Activer KM1
Logique de Commande de Moteur
Lire Capteurs
Condition à ao moteur?
No
Activer KM1
Ne
Activer KM1
Activer Moteur des moteos
Activer Moteur
Condition à ao moissen?
No
Mis
Activer KM2
Activer Moteur
Start
Logigramme / Algorigramme
Schéma Électrique de Puissance
K1
Q2
K3
Q1
Contoult Cirorit bresker oo 20 AC 19 300Nz
T1
Eanteurs transforme
Sonplorment de commande do 309 90Hz
Commoporise relaitee
KM1
KM2
Goltes cornefams modulles povaiant fles
Sinuplerment de hitraut de mosomenns à lire transfemes electrigos
M1 Motor
M2 Motor
Schéma Électrique de Puissance
Schéma de Câblage
X1
X2
Schéma de Câblage
Power Supplt
Senseurs Analog
Intrnls Analogue
Intrnls Analogue
Intrnls Digitaler
MCU
MCU Contrôleur
Intrdls Analogue
Digitals Output
Digitals Output
Commande UNIT
Power
Schéma d'Architecture de Circuit

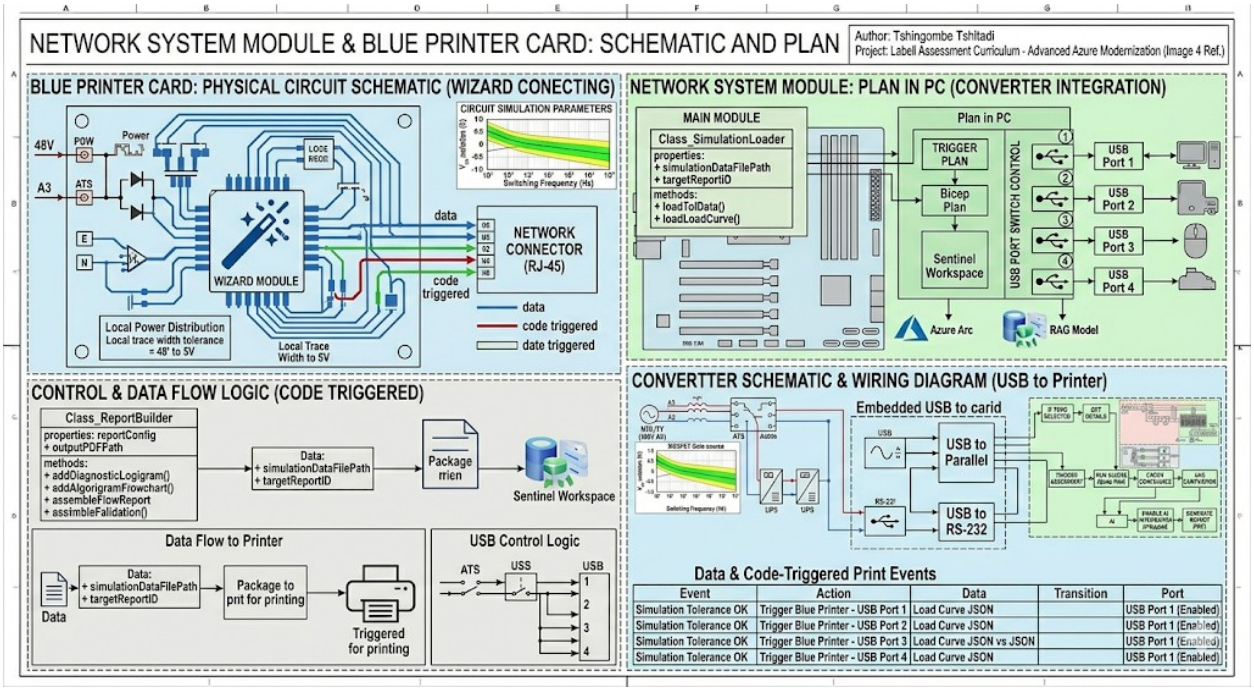

SYSTEM ARCHITECTURE: INTEGRATED ANALYSIS & GOVERNANCE
Project Explorer
VBAProject (Project1)
Electronics
Sheet1 (Sheet)
UserForm1
Software_Modules
Sheet1 (Sheets)
Sheet2(Etaol)
UserForm2
Analysis_Tools
Sheet3 (Sheets)
Sheet1 (Etaol)
Vbation Config
PhD_Integration
Network Linked Reposito
UserForm
ELECTRONICS ANALYSE CIRCUIT
PASSIVE CIRCUIT ANALYSIS
ACTIVE CIRCUIT ANALYSIS
ACTIVE CIRCUIT ANALYS
CIRCUIT OPTIMIZATION TASK
TASK CALL
TASK ORGANIGRAME
TASK CALL
TASK RESPONSE
MODULES & CLASSES
COMPULATION
GENERATIVE AI CURRICULUM OPTIMIZED
4IR Algorigramme
Synthesis Framework
4IR & Modernization Drivers
System Status: Active, Analyzing...
Synthesis Framework
Author: Tshingombe Tshitadi

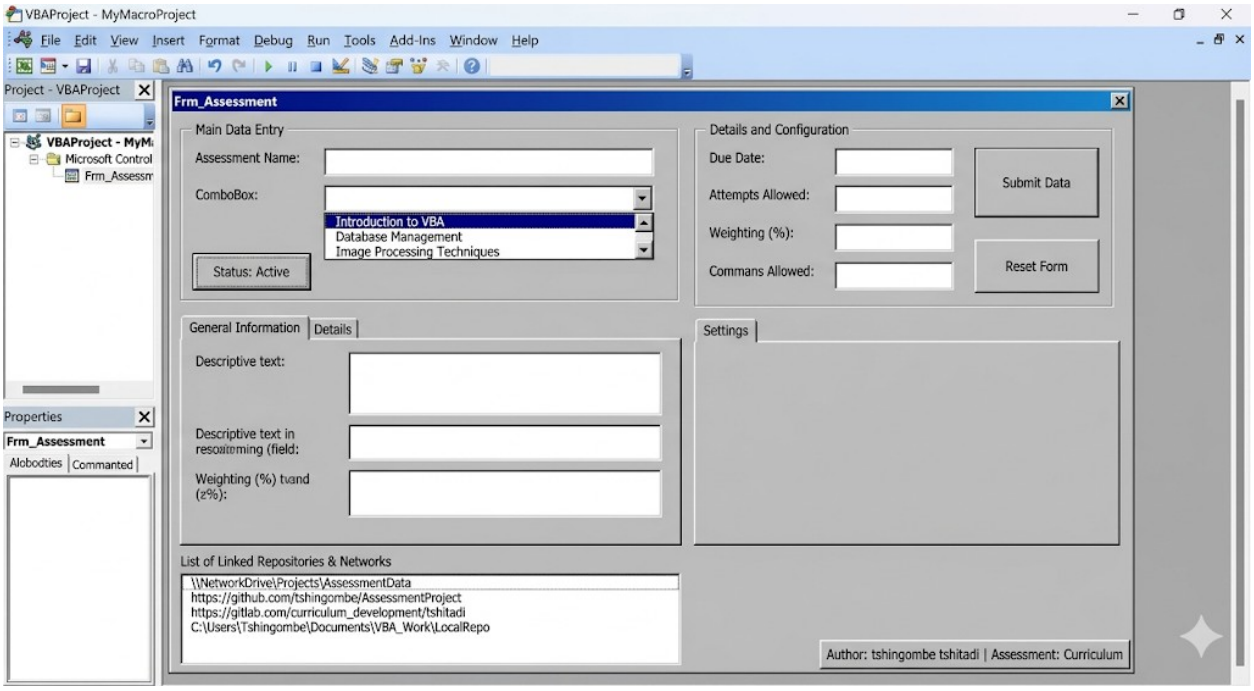

BLUE PRINTER CARD: WIZARD-LEVEL INTEGRATION & DATA PATH
INTEGRATION WIZARD FLOW
Wizard Connection to Wizard
BLUE PRINTER CARD
DATA CODE TRIGGERED FLOW
Software Events Triggereor FLOW
Switching Logic
SWITCH CONTROL
PC CONVERTER
(WIZARD_v3)
USB
PORT
SWITCH CONTROL
USB
NETWORK
NETWORK
SYSTEM MODULE
Network
NETWORK NTERFACE
NETWORK PORT (RJ45)
MODEM INTERFACE
Network IP Configuration
Schematic Ver: 3.2.1
System: Wizard-Connected
Date: 2026-05-19

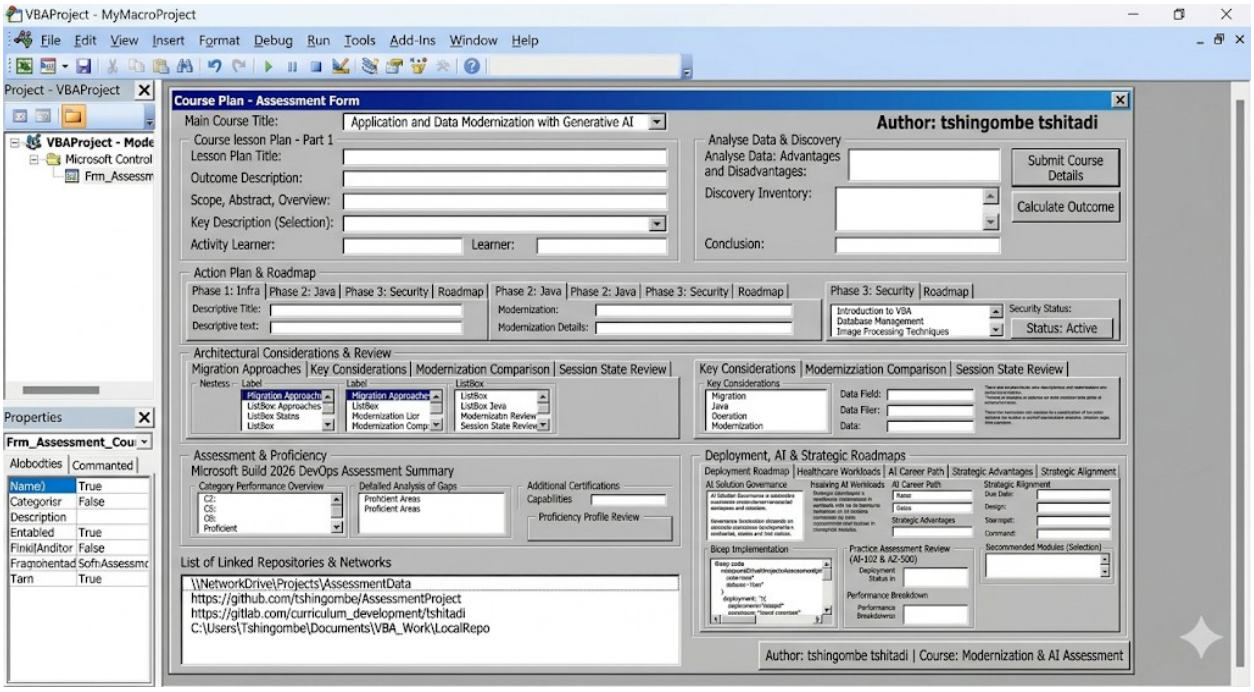

VBA Project [Labell Assessment Curriculum] - [Author: Tshingombe Tshitadi]
File Edit View Insert Format Debug Run Tools Add-ins Window Help
Ln 6, Col 1
Tshingombe Tshitadi - Assessment Curriculum
NETWORK REPOS
List of Linked Repos
https://github.com/Tshingombe/modernization-framework
azure://repos/cloud-native-iaas
gitlab://projects/ai-rdi-standards
Add Repo
Filter by Repo Type:
USER INPUTS
Project Title
Application and Data Modernization with Generative AI & Cloud Infrastructure
Abstract
pedagogical and technical framework for...
Modernization Approach
Refactoring (Cloud-Native)
Security Audit Enabled
Compliance Check
Performance Test
AI Integration
Run Assessment
Generate Report
ASSESSMENT SUMMARY
Network Isolation
Java Refactoring
Identity Sync
RAG Model Deployment
Private Endpoint Bicep configuration
NSGs configured on subnet snet-private-endpoints
Azure Bastion deployed


VBA Project [Labell Assessment Curriculum]
File Edit View Insert Format Debug Run Tools Add-ins Window Help
Project - VBA Project
Properties - VBAject
Tshingombe Tshitadi - Assessment Curriculum
NETWORK REPOS
List of Linked Repos
https://github.com/Tshingombe/modernization-framework
azure://repos/cloud-native-iaas
azure://repos/cloud-native-iaas
gitlab://projects/ai-rdi-standards
https://github.com/Tshingombe/modernization-framework
gitlab://projects/ai-rdi-standards
Add Repo
Filter by Repo Type
USER INPUTS
Project Title
Application and Data Modernization with Generative AI & Cloud Infrastructure
Abstract
Comprehensive pedagogical and technical framework for...
Security Audit Enabled
Compliance Check
Performance Test
AI Integration
Run Assessment
Generate Report
ASSESSMENT SUMMARY
Page
Network Isolation
Java Refactoring
Identity Sync
RAG Model Deployment
Private Isolation & private endpoints from azurk to endpoints existing private eveldion configurations:
- Private endpoints to be complicated to NSGs:
- Azure Bastion configurations to Azure HP rool Bastion.


SYSTEM MODERNIZATION FRAMEWORK: SCHEMATIC AND LOGIC
Author: Tshingombe Tshitadi
Project: Labell Assessment Curriculum - Azure Modernization (Image 2 Ref.)
SCHEMATIC ELECTRICAL POWER (SCHEMA PUISSANCE)
UTILITY (400V AC)
ATS
Automatic Transfer Switch
UPS
SERVER CLUSTER - MODERNIZED
LOGIGRAMME (LOGIC FLOW)
IMAGE_2.PNG UI INTERFACE
IF REPO SELECTED
GET DETAILS
SELECTE
Bicep Plan
TRIGGER
Sentinel Workspace
TRIGGER ASSESSMENT
RUN ASSESS (Bicep Plan)
CHECK COMPLIANCE
AUG COMPUATION
AI
ENABLE AI INTEGRATION (if toggled)
GENERATE REPORT (PDF)
Azure Arc
RAG Model
WIRING DIAGRAM (SCHEMA CABLAGE)
CONTROL PANEL
REPOS REPOSITORY
AI FOUNDRY
Bicep
sentinel
Sentinel Workspace
GATEWAY
ALGORIGRAMME (FLOWCHART)
RUN ASSESSMENT
START
CHECK INTERNET CONN
IF NOT
ALERT (Bastion needed?)
IF YES
PULL BICEP TEMPLATE
VALIDATE RESOURCES
LOOP THROUGH REGIONS (Azure HP root Bastion, in-nest-azures)
CHECK NSG CONFIGURATION
SUMMARIZE GAPS
IF GAPS
NO
LOG RISK
YES
OUTPUT RESULT
CONTROL CIRCUIT (CIRCUIT DE COMMANDE)
MAIN UI STATU
INTEGRATED 4IR ARCHITECTURAL FRAMEWORK: CIRCUIT & AUTOMATION SYNERGY
Author: Tshingombe Tshitadi
Project: Labell Assessment Curriculum - Advanced Azure Modernization (Image 4 Ref.)
CIRCUIT SIMULATION PARAMETERS (DETAIL - MONITOR DRIVE)
MOSFET Gate-source
Switching Frequency (Hz)
VBA AUTOMATION ENGINE (BACKEND CLASS STRUCTURES)
MAIN MODULE
Class_SimulationLoader
properties:
+ simulationDataFilePath
+ targetReportID
methods:
+ loadToIData()
+ loadLoadCurve()
Class_AssessmentCalculator
properties:
+ inputScoreVector
+ referenceComplianceLevel
methods:
+ runGapAnalysis()
+ runBicepValidation()
Class_ReportBuilder
properties: reportConfig
+ outputPDFPath
methods:
+ addDiagnosticLogigram()
+ addAlgorigramFlowchart()
+ assembleFinalReport
Class_LogigramGenerator
properties: logicalInputs
+ logicRules
methods:
+ drawGates()
+ createFlowPath()
Class_AlgorigramGenerator
properties: programFlowLogic
- loopConditions
methods:
+ createFlowchartElements()
+ addDecisionPoints()
Execution Routine Lifecycle
Class_SimulationLoader
MAIN MODULE
Class_AssessmentCalculator
Class_ReportBuilder
Class_LogigramGenerator
Class_AlgorigramGenerator
Class_AlgorigramGenerator
Integrated Diagnostic Report (PDF)
Execution Routine Lifecycle
Passive Filter Performance (LC)
Ripple Rejection (%)
Component Variance (L± 10%)
Operational Load Curves
Parameter
Power (W)
Peak Temp.
Bandwidth (Gbps)
4IR Workload
Workload 3


NETWORK SYSTEM MODULE & BLUE PRINTER CARD: SCHEMATIC AND PLAN
Author: Tshingombe Tshitadi
Project: Labell Assessment Curriculum - Advanced Azure Modernization (Image 4 Ref.)
BLUE PRINTER CARD: PHYSICAL CIRCUIT SCHEMATIC (WIZARD CONECTING)
CIRCUIT SIMULATION PARAMETERS
48V
A3
POW
ATS
E
N
Power
WIZARD MODULE
data
code triggered
NETWORK CONNECTOR (RJ-45)
data
code triggered
date triggered
Local Power Distribution
Local trace width tolerance = 48' to 5V
Local Trace Width to 5V
NETWORK SYSTEM MODULE: PLAN IN PC (CONVERTER INTEGRATION)
MAIN MODULE
Class_SimulationLoader
properties:
+ simulationDataFilePath
+ targetReportIO
methods:
+ loadToIData()
+ loadLoadCurve()
Plan in PC
TRIGGER PLAN
Bicep Plan
Sentinel Workspace
USB PORT SWITCH CONTROL
USB Port 1
USB Port 2
USB Port 3
USB Port 4
Azure Arc
RAG Model
CONTROL & DATA FLOW LOGIC (CODE TRIGGERED)
Class_ReportBuilder
properties: reportConfig
+ outputPDFPath
methods:
+ addDiagnosticLogigram()
+ addAlgorigramFrowchart()
+ assembleFlowReport
+ assimbleFallidation()
Data:
+ simulationDataFilePath
+ targetReportID
Package rrien
Sentinel Workspace
Data Flow to Printer
Data
Data:
+ simulationDataFilePath
+ targetReportID
Package to pnt for printing
Triggered for printing
USB Control Logic
ATS
USS
USB
1
2
3
4
CONVERTTER SCHEMATIC & WIRING DIAGRAM (USB to Printer)
Embedded USB to carid
USB to Parallel
USB to RS-232
Data & Code-Triggered Print Events
Event
Action
Data
Transition
Port
Simulation Tolerance OK
Trigger Blue Printer - USB Port 1
Load Curve JSON
USB Port 1 (Enabled)
Simulation Tolerance OK
Trigger Blue Printer - USB Port 2
Load Curve JSON
USB Port 1 (Enabled)
Simulation Tolerance OK
Trigger Blue Printer - USB Port 3
Load Curve JSON vs JSON
USB Port 1 (Enabled)
Simulation Tolerance OK
Trigger Blue Printer - USB Port 4
Load Curve JSON
USB Port 1 (Enabled)


VBAProject - MyMacroProject
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Project - VBAProject
VBAProject - MyM
Microsoft Control
Frm_Assessm
Properties
Frm_Assessment
Alobodties
Commanted
Frm_Assessment
Main Data Entry
Assessment Name:
ComboBox:
Introduction to VBA
Database Management
Image Processing Techniques
Status: Active
Details and Configuration
Due Date:
Attempts Allowed:
Weighting (%):
Commans Allowed:
Submit Data
Reset Form
General Information
Details
Descriptive text:
Descriptive text in
Weighting (%) tvand
(z%):
Settings
List of Linked Repositories & Networks
\\NetworkDrive\Projects\AssessmentData
https://github.com/tshingombe/AssessmentProject
https://gitlab.com/curriculum_development/tshitadi
C:\Users\Tshingombe\Documents\VBA_Work\LocalRepo
Author: tshingombe tshitadi | Assessment: Curriculum


VBAProject - MyMacroProject
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Project - VBAProject
Properties
Course Plan - Assessment Form
Main Course Title:
Application and Data Modernization with Generative AI
Author: tshingombe tshitadi
Course lesson Plan - Part 1
Lesson Plan Title:
Outcome Description:
Scope, Abstract, Overview:
Key Description (Selection):
Activity Learner:
Learner:
Analyse Data & Discovery
Analyse Data: Advantages and Disadvantages:
Discovery Inventory:
Conclusion:
Submit Course Details
Calculate Outcome
Action Plan & Roadmap
Phase 1: Infra
Phase 2: Java
Phase 3: Security
Roadmap
Phase 2: Java
Phase 2: Java
Phase 3: Security
Roadmap
Phase 3: Security
Roadmap
Descriptive Title:
Descriptive text:
Modernization:
Modernization Details:
Introduction to VBA
Database Management
Image Processing Techniques
Security Status:
Status: Active
Architectural Considerations & Review
Migration Approaches
Key Considerations
Modernization Comparison
Session State Review
Key Considerations
Modernizziation Comparison
Session State Review
Key Considerations
Migration
Java
Operation
Modernization
Data Field:
Data Filer:
Data:
Assessment & Proficiency
Microsoft Build 2026 DevOps Assessment Summary
Category Performance Overview
Detailed Analysis of Gaps
Additional Certifications
Capabilities
Proficiency Profile Review
Deployment, AI & Strategic Roadmaps
Deployment Roadmap
Healthcare Workloads
AI Career Path
Strategic Advantages
Strategic Alignment
AI Solution Governance
Insaiving AI Workloads
AI Career Path
Strategic Alignment
Strategic Advantages
Bicep Implementation
Practice Assessment Review (AI-102 & AZ-500)
Performance Breakdown
Secommended Modules (Selection)
List of Linked Repositories & Networks
\\NetworkDrive\Projects\AssessmentData
https://github.com/tshingombe/AssessmentProject
https://gitlab.com/curriculum_development/tshitadi
C:\Users\Tshingombe\Documents\VBA_Work\LocalRepo
Author: tshingombe tshitadi | Course: Modernization & AI Assessment

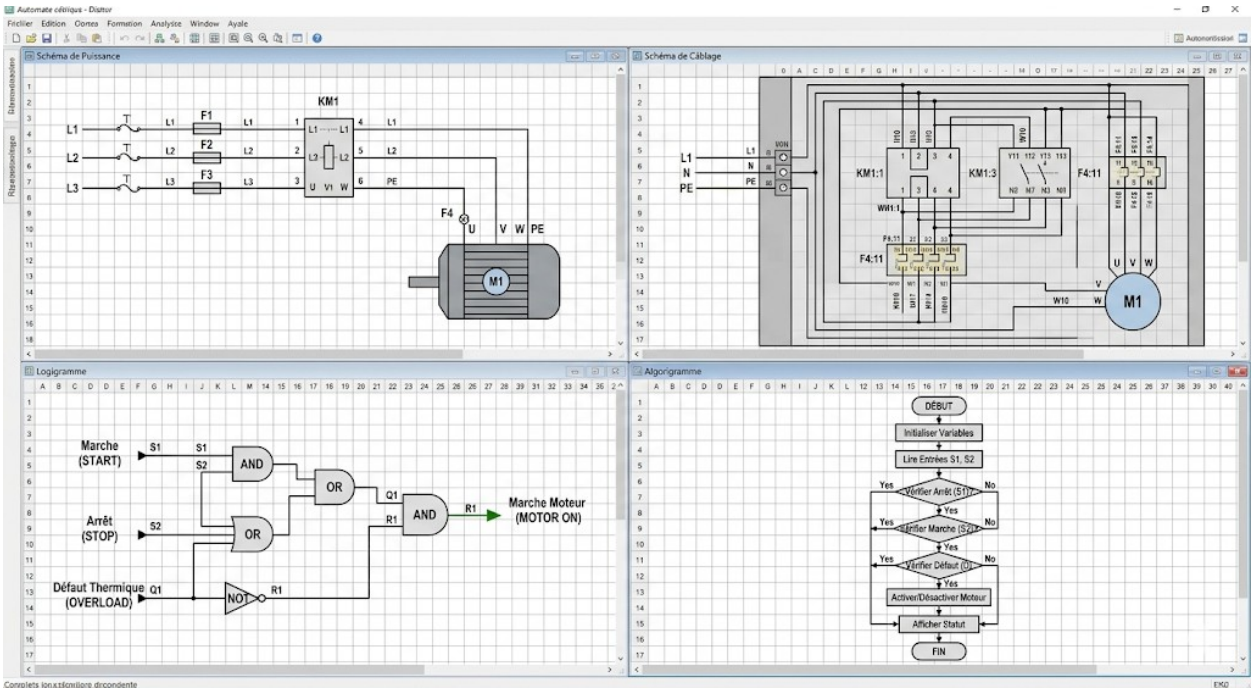


Schéma de Puissance
Schéma de Câblage
Logigramme
Algorigramme
KM1
F1
F2
F3
F4
M1
L1
L2
L3
PE
Marche (START)
Arrêt (STOP)
Défaut Thermique (OVERLOAD)
AND
OR
NOT
Marche Moteur (MOTOR ON)
DÉBUT
Initialiser Variables
Lire Entrées S1, S2
Vérifier Arrêt (S1)
Vérifier Marche (S2)
Vérifier Défaut (Q1)
Activer/Désactiver Moteur
Afficher Statut
FIN

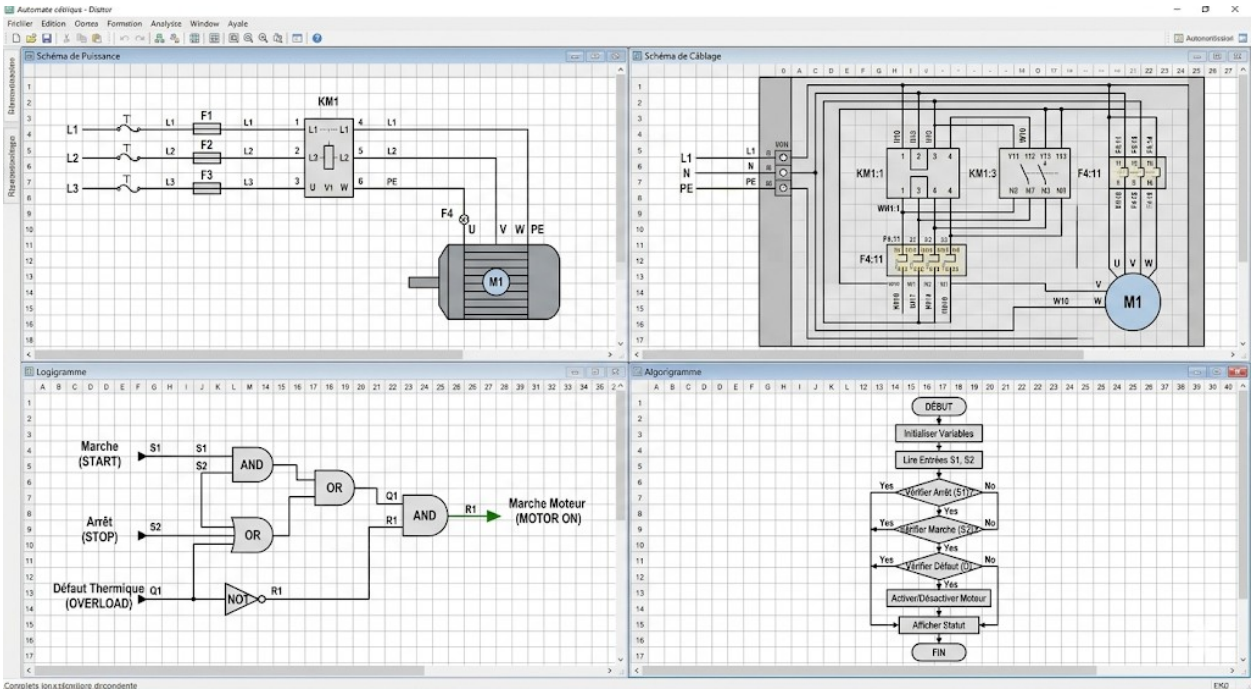
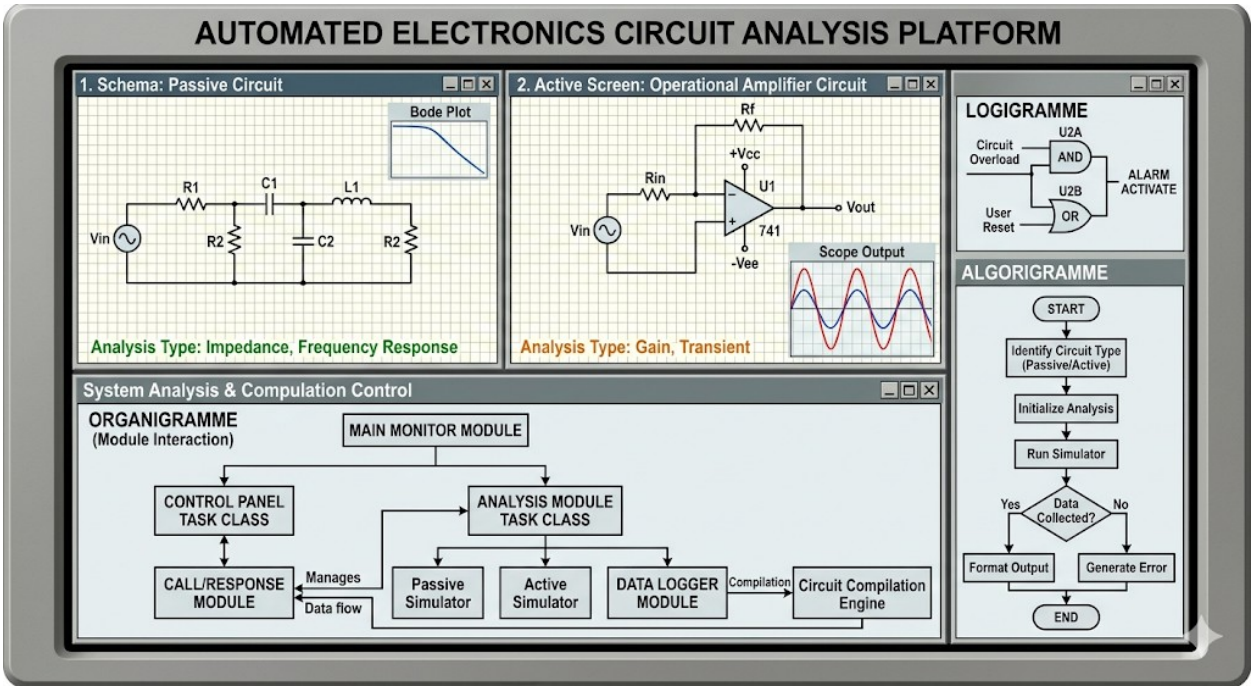
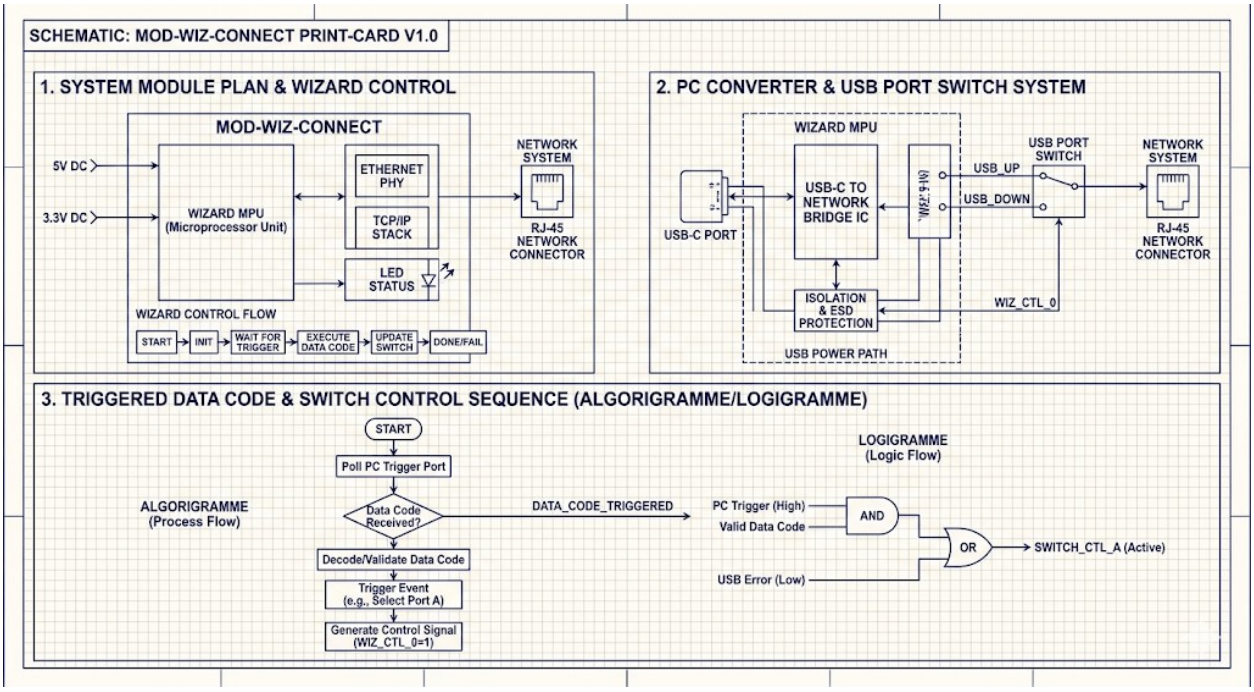
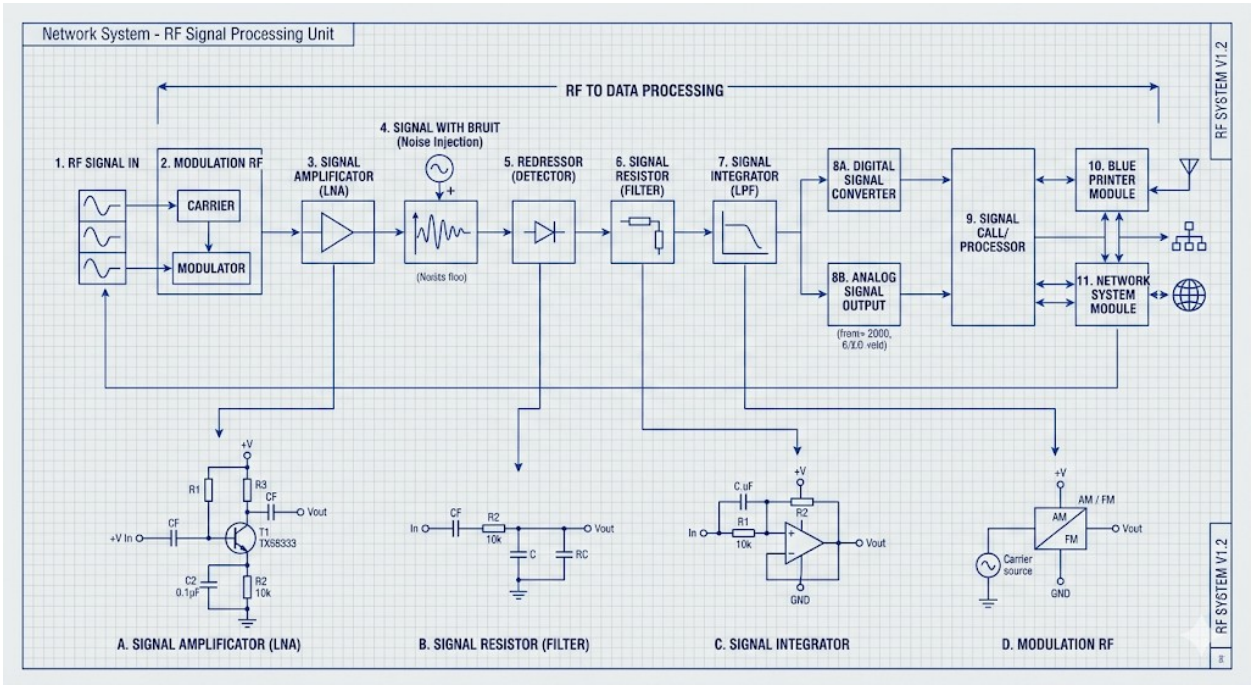
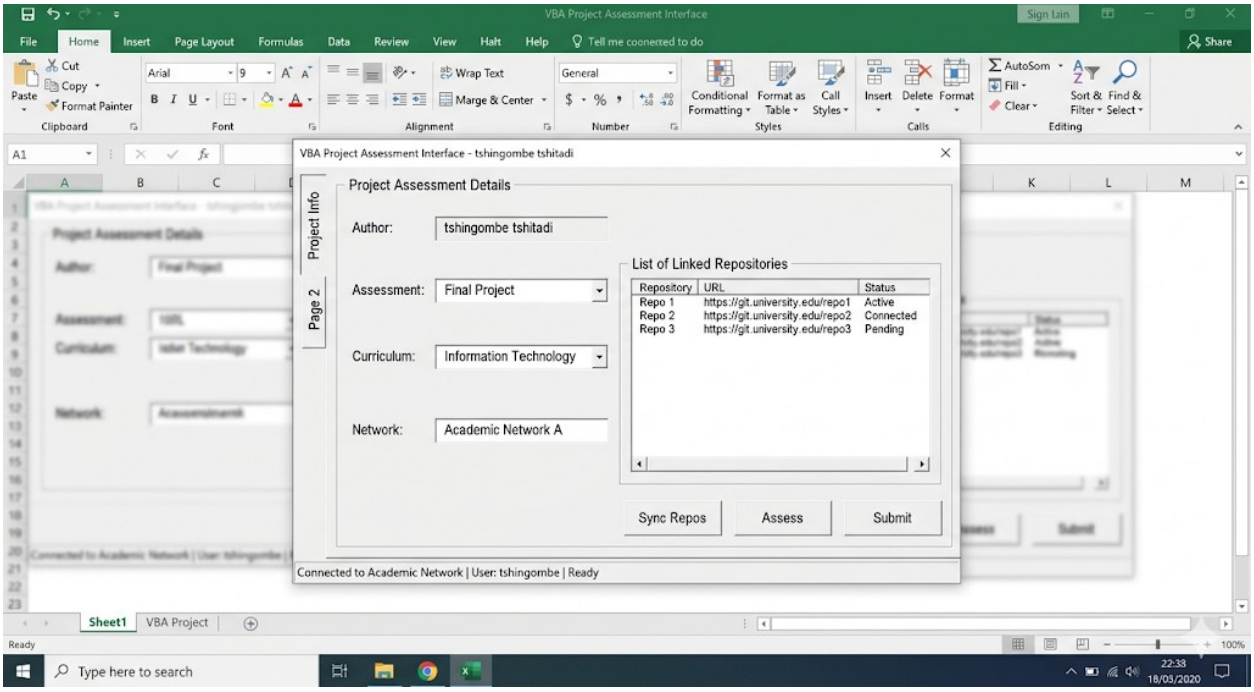


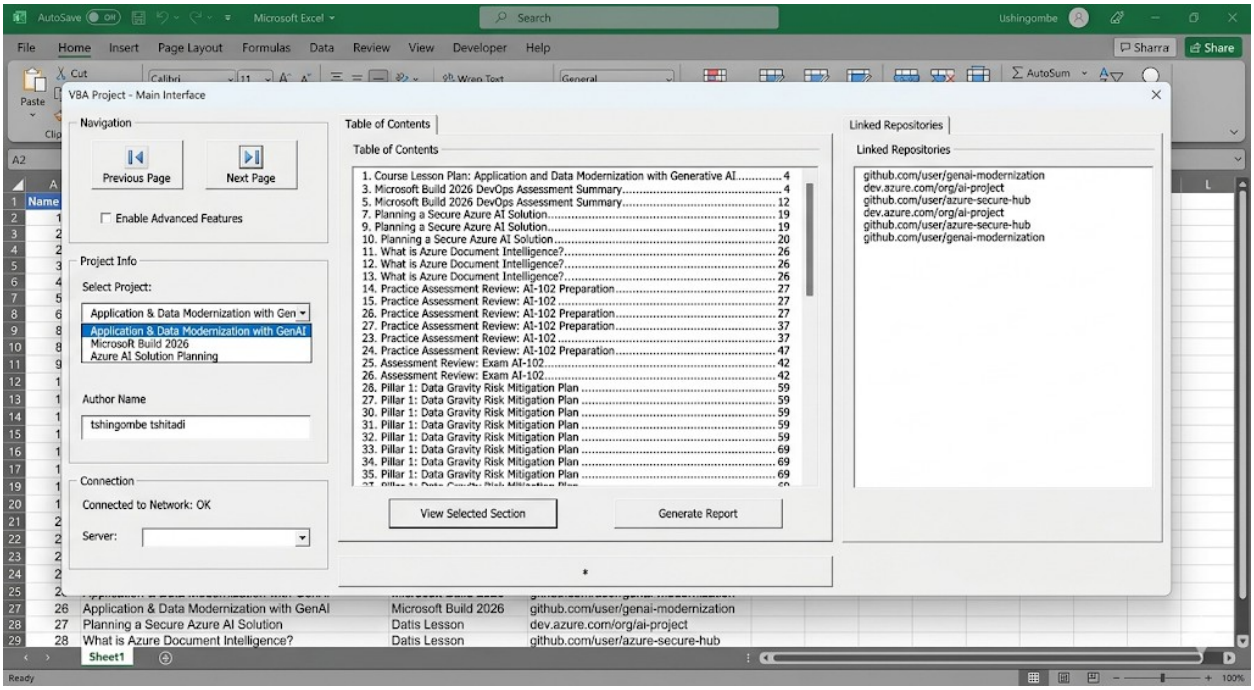
Schéma de Puissance
KM1
F1
F2
F3
F4
L1
L2
L3
PE
U V W
M1
Schéma de Câblage
KM1:1
KM1:3
F4:11
M1
Logigramme
Marche (START)
Arrêt (STOP)
Défaut Thermique (OVERLOAD)
AND
OR
NOT
Marche Moteur (MOTOR ON)
Algorigramme
DÉBUT
Initialiser Variables
Lire Entrées S1, S2
Vérifier Arrêt (S1)?
Vérifier Marche (S2)?
Vérifier Défaut (Q)?
Activer/Désactiver Moteur
Afficher Statut
FIN

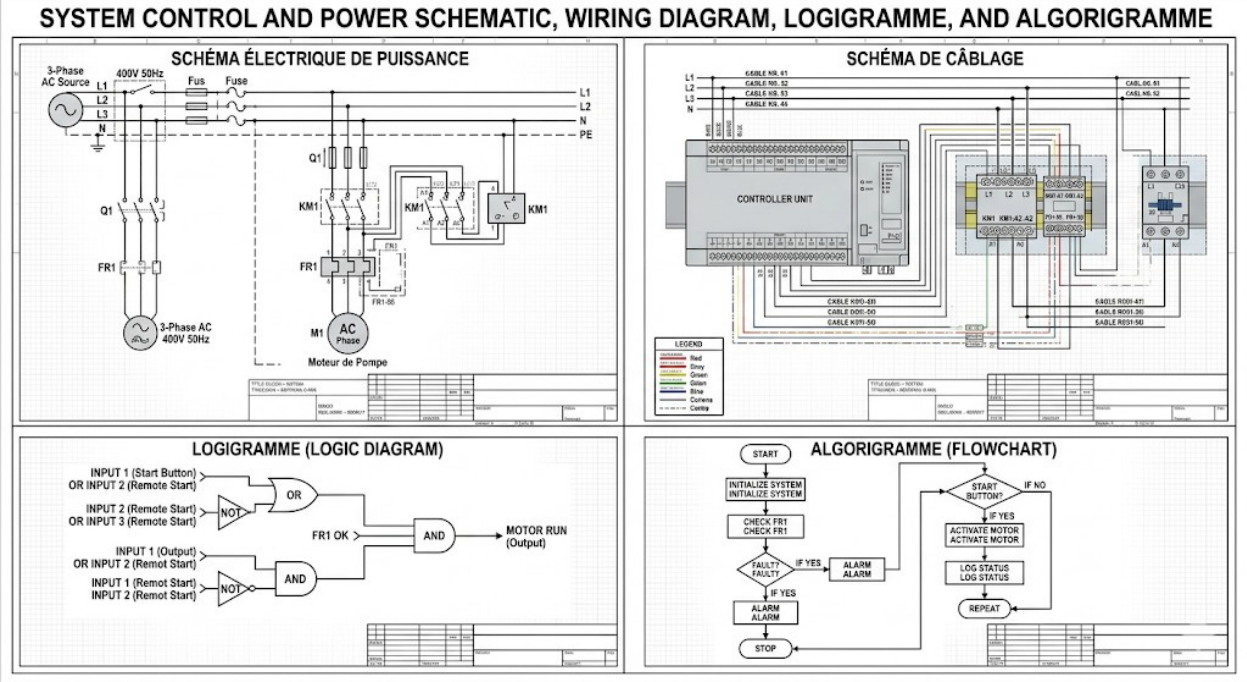

AUTOMATED ELECTRONICS CIRCUIT ANALYSIS PLATFORM
1. Schema: Passive Circuit
Bode Plot
R1
C1
L1
R2
C2
R2
Vin
Analysis Type: Impedance, Frequency Response
2. Active Screen: Operational Amplifier Circuit
Rf
+Vcc
Rin
U1
Vout
Vin
741
-Vee
Scope Output
Analysis Type: Gain, Transient
LOGIGRAMME
Circuit Overload
U2A
AND
ALARM ACTIVATE
U2B
User Reset
OR
ALGORIGRAMME
START
Identify Circuit Type (Passive/Active)
Initialize Analysis
Run Simulator
Data Collected?
Yes
No
Format Output
Generate Error
END
System Analysis & Compulation Control
ORGANIGRAMME
(Module Interaction)
MAIN MONITOR MODULE
CONTROL PANEL TASK CLASS
ANALYSIS MODULE TASK CLASS
CALL/RESPONSE MODULE
Manages
Data flow
Passive Simulator
Active Simulator
DATA LOGGER MODULE
Compilation
Circuit Compilation Engine

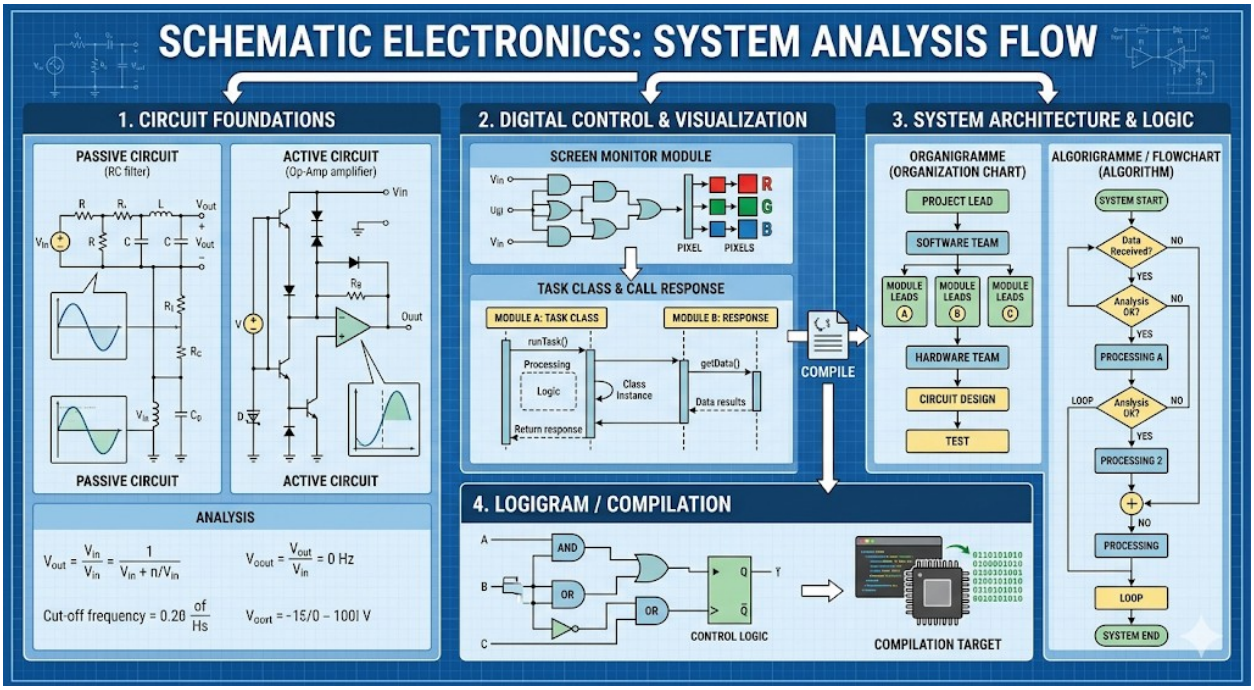

SCHEMATIC: MOD-WIZ-CONNECT PRINT-CARD V1.0
1. SYSTEM MODULE PLAN & WIZARD CONTROL
MOD-WIZ-CONNECT
5V DC
3.3V DC
WIZARD MPU (Microprocessor Unit)
ETHERNET PHY
TCP/IP STACK
LED STATUS
NETWORK SYSTEM
RJ-45 NETWORK CONNECTOR
WIZARD CONTROL FLOW
START
INIT
WAIT FOR TRIGGER
EXECUTE DATA CODE
UPDATE SWITCH
DONE/FAIL
2. PC CONVERTER & USB PORT SWITCH SYSTEM
WIZARD MPU
USB-C PORT
USB-C TO NETWORK BRIDGE IC
USB_UP
USB_DOWN
USB PORT SWITCH
NETWORK SYSTEM
RJ-45 NETWORK CONNECTOR
ISOLATION & ESD PROTECTION
WIZ_CTL_0
USB POWER PATH
3. TRIGGERED DATA CODE & SWITCH CONTROL SEQUENCE (ALGORIGRAMME/LOGIGRAMME)
ALGORIGRAMME (Process Flow)
START
Poll PC Trigger Port
Data Code Received?
DATA_CODE_TRIGGERED
Decode/Validate Data Code
Trigger Event (e.g., Select Port A)
Generate Control Signal (WIZ_CTL_0=1)
LOGIGRAMME (Logic Flow)
PC Trigger (High)
Valid Data Code
AND
OR
SWITCH_CTL_A (Active)
USB Error (Low)

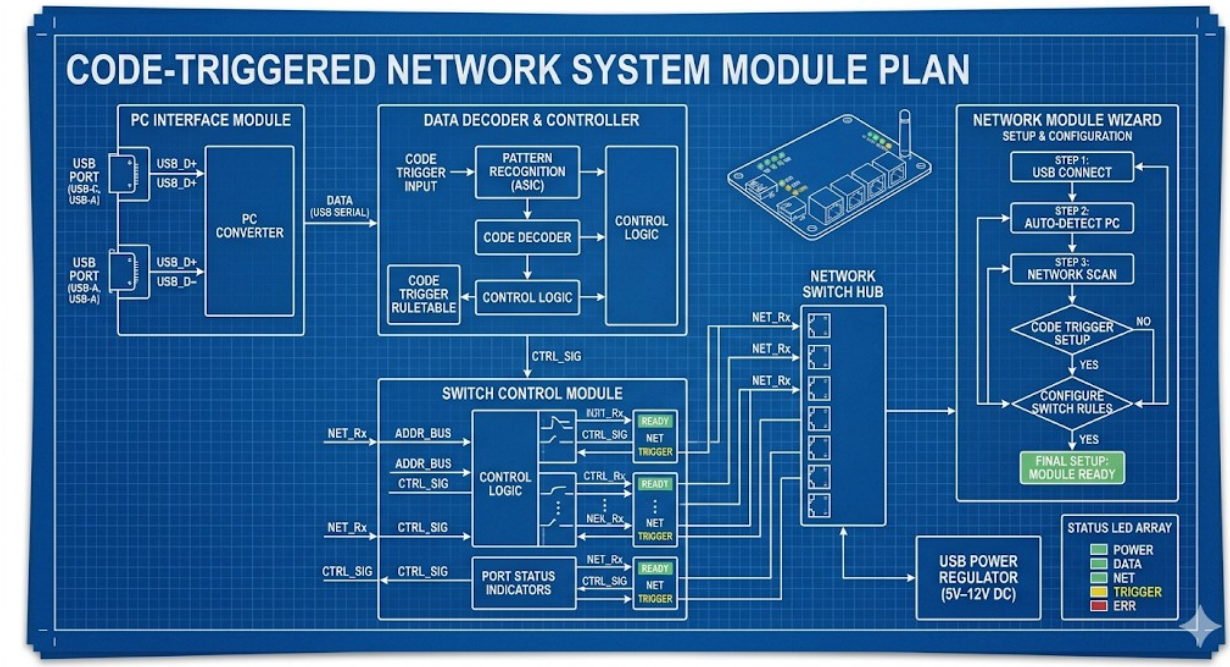

Network System - RF Signal Processing Unit
RF TO DATA PROCESSING
RF SYSTEM V1.2
1. RF SIGNAL IN
2. MODULATION RF
CARRIER
MODULATOR
3. SIGNAL AMPLIFICATOR (LNA)
4. SIGNAL WITH BRUIT (Noise Injection)
5. REDRESSOR (DETECTOR)
6. SIGNAL RESISTOR (FILTER)
7. SIGNAL INTEGRATOR (LPF)
8A. DIGITAL SIGNAL CONVERTER
8B. ANALOG SIGNAL OUTPUT
9. SIGNAL CALL/ PROCESSOR
10. BLUE PRINTER MODULE
11. NETWORK SYSTEM MODULE
A. SIGNAL AMPLIFICATOR (LNA)
B. SIGNAL RESISTOR (FILTER)
C. SIGNAL INTEGRATOR
D. MODULATION RF
AM / FM
Carrier source
GND
Vout

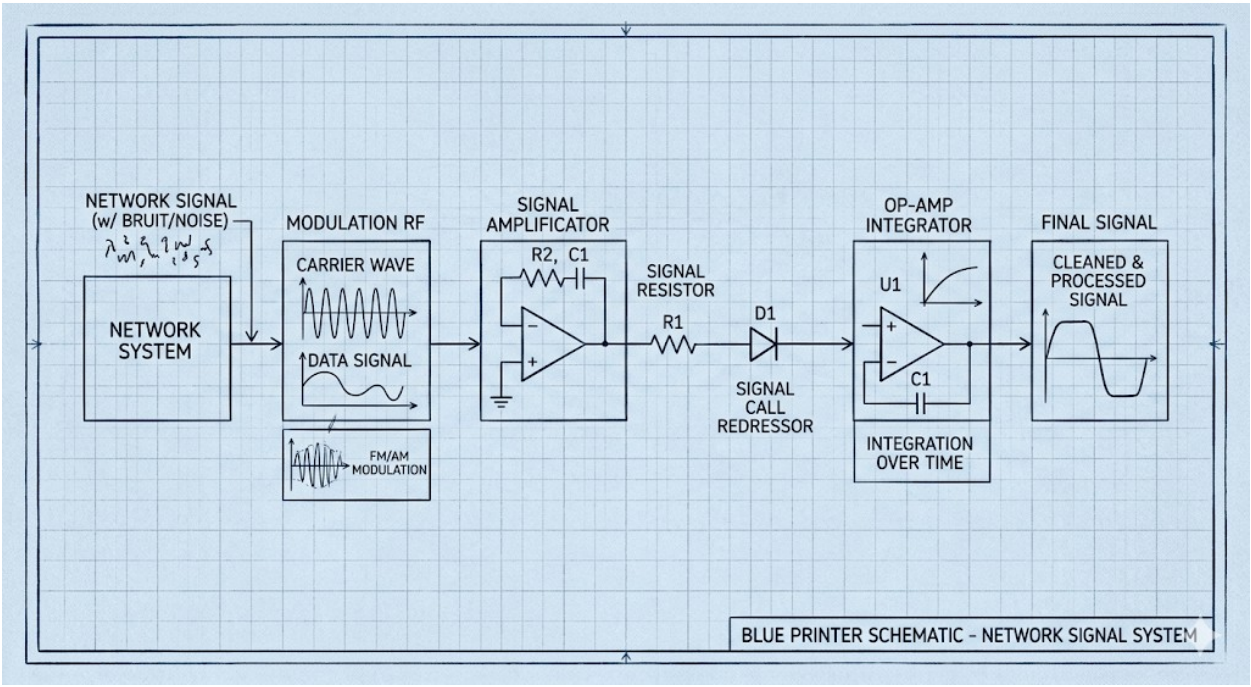

VBA Project Assessment Interface - tshingombe tshitadi
Project Info
Page 2
Project Assessment Details
Author:
tshingombe tshitadi
Assessment:
Final Project
Curriculum:
Information Technology
Network:
Academic Network A
List of Linked Repositories
Repository
URL
Status
Repo 1
https://git.university.edu/repo1
Active
Repo 2
https://git.university.edu/repo2
Connected
Repo 3
https://git.university.edu/repo3
Pending
Sync Repos
Assess
Submit
Connected to Academic Network | User: tshingombe | Ready


VBA Project - Main Interface
Navigation
Previous Page
Next Page
Enable Advanced Features
Project Info
Select Project:
Application & Data Modernization with GenAI
Microsoft Build 2026
Azure AI Solution Planning
Author Name
tshingombe tshitadi
Connection
Connected to Network: OK
Server:
Table of Contents
Linked Repositories
View Selected Section
Generate Report


SYSTEM CONTROL AND POWER SCHEMATIC, WIRING DIAGRAM, LOGIGRAMME, AND ALGORIGRAMME
SCHÉMA ÉLECTRIQUE DE PUISSANCE
3-Phase AC Source
400V 50Hz
L1
L2
L3
N
PE
Fus
Fuse
Q1
KM1
FR1
3-Phase AC 400V 50Hz
M1
AC Phase
Moteur de Pompe
SCHÉMA DE CÂBLAGE
CONTROLLER UNIT
LEGEND
LOGIGRAMME (LOGIC DIAGRAM)
INPUT 1 (Start Button)
OR INPUT 2 (Remote Start)
INPUT 2 (Remote Start)
OR INPUT 3 (Remote Start)
INPUT 1 (Output)
OR INPUT 2 (Remot Start)
INPUT 1 (Remot Start)
INPUT 2 (Remot Start)
NOT
OR
AND
FR1 OK
MOTOR RUN (Output)
ALGORIGRAMME (FLOWCHART)
START
INITIALIZE SYSTEM
INITIALIZE SYSTEM
CHECK FR1
CHECK FR1
FAULT? FAULT?
IF YES
ALARM
ALARM
START BUTTON?
IF NO
ACTIVATE MOTOR
ACTIVATE MOTOR
LOG STATUS
LOG STATUS
REPEAT
STOP


SCHEMATIC ELECTRONICS: SYSTEM ANALYSIS FLOW
1. CIRCUIT FOUNDATIONS
PASSIVE CIRCUIT
(RC filter)
ACTIVE CIRCUIT
(Op-Amp amplifier)
PASSIVE CIRCUIT
ACTIVE CIRCUIT
ANALYSIS
Cut-off frequency = 0.20 of Hs
2. DIGITAL CONTROL & VISUALIZATION
SCREEN MONITOR MODULE
PIXEL
PIXELS
TASK CLASS & CALL RESPONSE
MODULE A: TASK CLASS
MODULE B: RESPONSE
runTask()
Processing
Logic
Return response
Class Instance
getData()
Data results
COMPILE
3. SYSTEM ARCHITECTURE & LOGIC
ORGANIGRAMME (ORGANIZATION CHART)
PROJECT LEAD
SOFTWARE TEAM
MODULE LEADS
HARDWARE TEAM
CIRCUIT DESIGN
TEST
ALGORIGRAMME / FLOWCHART (ALGORITHM)
SYSTEM START
Data Received?
Analysis OK?
PROCESSING A
PROCESSING 2
PROCESSING
LOOP
SYSTEM END
4. LOGIGRAM / COMPILATION
AND
OR
CONTROL LOGIC
COMPILATION TARGET


CODE-TRIGGERED NETWORK SYSTEM MODULE PLAN
PC INTERFACE MODULE
USB PORT (USB-C, USB-A)
USB PORT (USB-A, USB-A)
USB_D+
USB_D+
USB_D+
USB_D−
PC CONVERTER
DATA (USB SERIAL)
DATA DECODER & CONTROLLER
CODE TRIGGER INPUT
PATTERN RECOGNITION (ASIC)
CODE DECODER
CONTROL LOGIC
CODE TRIGGER RULETABLE
CONTROL LOGIC
CTRL_SIG
SWITCH CONTROL MODULE
NET_Rx
ADDR_BUS
ADDR_BUS
CTRL_SIG
NET_Rx
CTRL_SIG
CTRL_SIG
CTRL_SIG
CONTROL LOGIC
PORT STATUS INDICATORS
INPT_Rx
CTRL_SIG
READY
NET TRIGGER
CTRL_Rx
READY
NET TRIGGER
NET_Rx
NET_Rx
CTRL_SIG
READY
NET TRIGGER
NETWORK SWITCH HUB
NET_Rx
NET_Rx
NET_Rx
NETWORK MODULE WIZARD
SETUP & CONFIGURATION
STEP 1: USB CONNECT
STEP 2: AUTO-DETECT PC
STEP 3: NETWORK SCAN
CODE TRIGGER SETUP
NO
YES
CONFIGURE SWITCH RULES
YES
FINAL SETUP: MODULE READY
USB POWER REGULATOR (5V–12V DC)
STATUS LED ARRAY
POWER
DATA
NET
TRIGGER
ERR


NETWORK SIGNAL (w/ BRUIT/NOISE)
NETWORK SYSTEM
MODULATION RF
CARRIER WAVE
DATA SIGNAL
FM/AM MODULATION
SIGNAL AMPLIFICATOR
R2, C1
SIGNAL RESISTOR
R1
D1
SIGNAL CALL REDRESSOR
OP-AMP INTEGRATOR
U1
C1
INTEGRATION OVER TIME
FINAL SIGNAL
CLEANED & PROCESSED SIGNAL
BLUE PRINTER SCHEMATIC - NETWORK SIGNAL SYSTEM

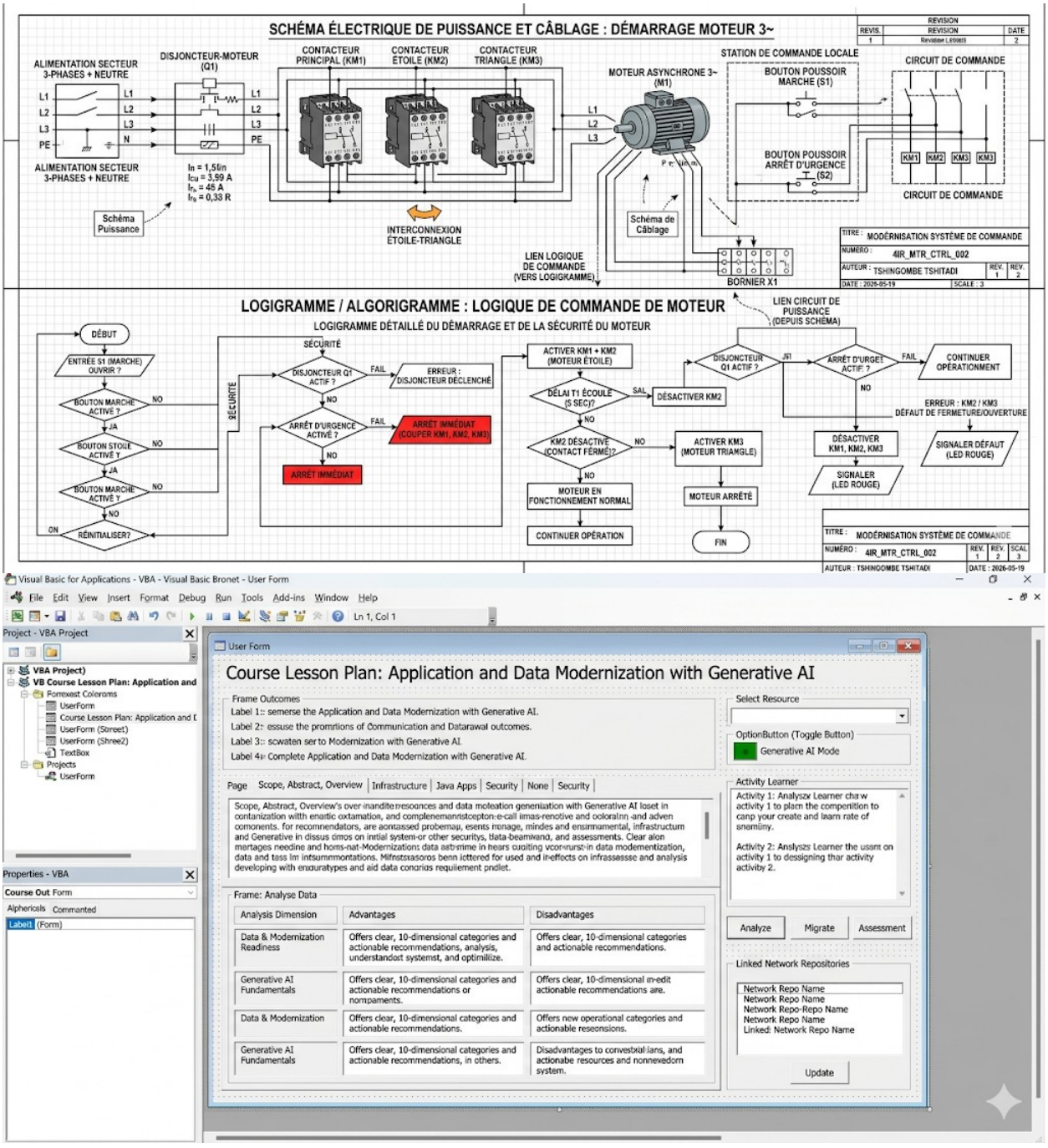
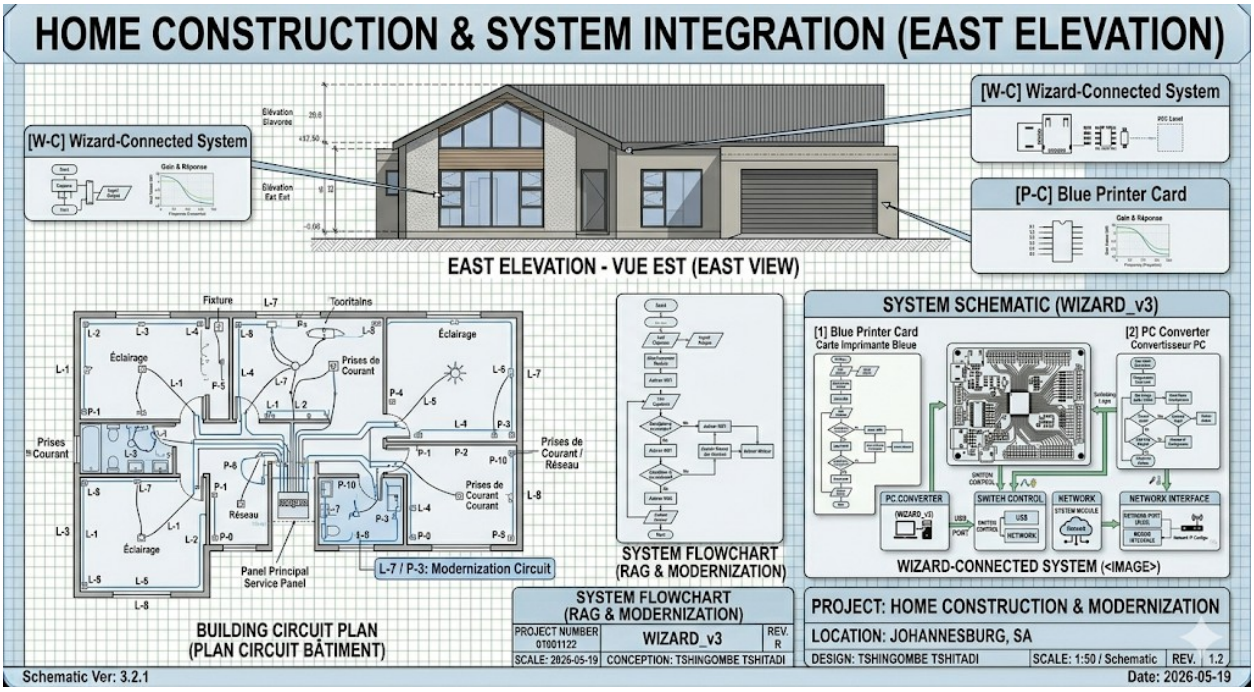
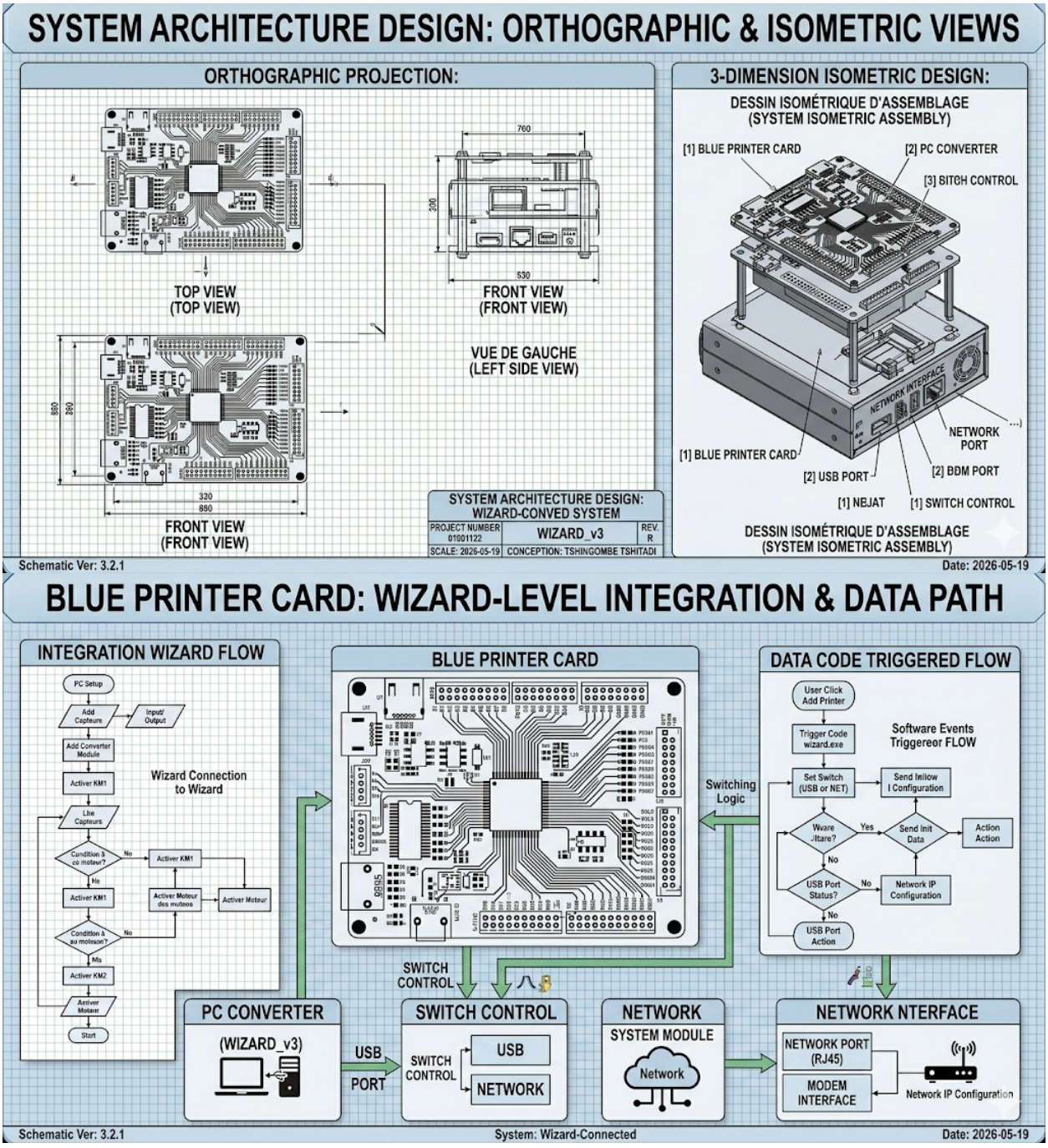


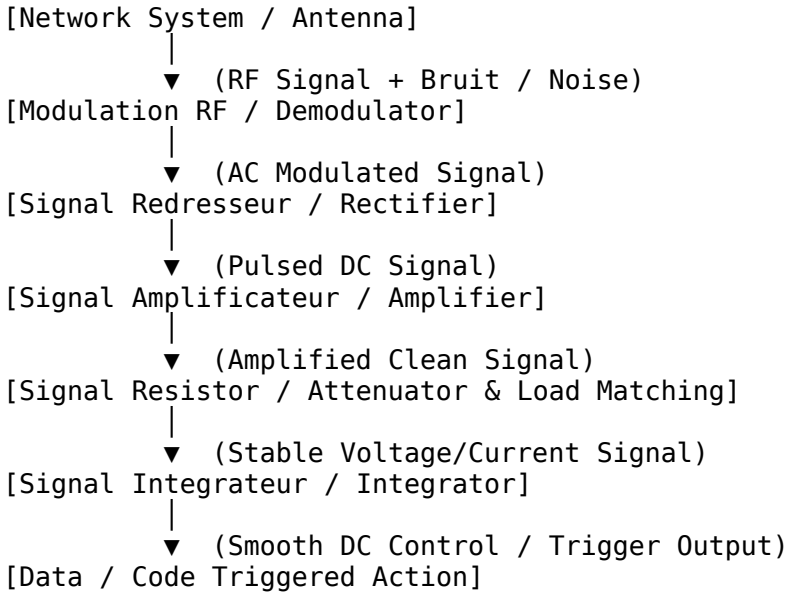
SCHÉMA ÉLECTRIQUE DE PUISSANCE ET CÂBLAGE : DÉMARRAGE MOTEUR 3~
ALIMENTATION SECTEUR 3-PHASES + NEUTRE
DISJONCTEUR-MOTEUR (Q1)
CONTACTEUR PRINCIPAL (KM1)
CONTACTEUR ÉTOILE (KM2)
CONTACTEUR TRIANGLE (KM3)
MOTEUR ASYNCHRONE 3~ (M1)
STATION DE COMMANDE LOCALE
BOUTON POUSSOIR MARCHE (S1)
BOUTON POUSSOIR ARRÊT D'URGENCE (S2)
CIRCUIT DE COMMANDE
INTERCONNEXION ÉTOILE-TRIANGLE
LIEN LOGIQUE DE COMMANDE (VERS LOGIGRAMME)
BORNIER X1
MODÉRNISATION SYSTÈME DE COMMANDE
4IR_MTR_CTRL_002
TSHINGOMBE TSHITADI
LOGIGRAMME / ALGORIGRAMME : LOGIQUE DE COMMANDE DE MOTEUR
LOGIGRAMME DÉTAILLÉ DU DÉMARRAGE ET DE LA SÉCURITÉ DU MOTEUR
LIEN CIRCUIT DE PUISSANCE (DEPUIS SCHÉMA)
DÉBUT
ENTRÉE S1 (MARCHE) OUVRIR ?
BOUTON MARCHE ACTIVÉ ?
BOUTON STOLE ACTIVÉ ?
BOUTON MARCHE ACTIVÉ ?
RÉINITIALISER?
SÉCURITÉ
DISJONCTEUR Q1 ACTIF ?
ERREUR : DISJONCTEUR DÉCLENCHÉ
ARRÊT D'URGENCE ACTIVÉ ?
ARRÊT IMMÉDIAT (COUPER KM1, KM2, KM3)
ARRÊT IMMÉDIAT
ACTIVER KM1 + KM2 (MOTEUR ÉTOILE)
DÉLAI T1 ÉCOULÉ (5 SEC)?
DÉSACTIVER KM2
KM2 DÉSACTIVÉ (CONTACT FERMÉ)?
ACTIVER KM3 (MOTEUR TRIANGLE)
MOTEUR EN FONCTIONNEMENT NORMAL
MOTEUR ARRÊTÉ
CONTINUER OPÉRATION
FIN
DISJONCTEUR Q1 ACTIF ?
ARRÊT D'URGE? ACTIF ?
CONTINUER OPÉRATIONMENT
ERREUR : KM2 / KM3 DÉFAUT DE FERMETURE/OUVERTURE
DÉSACTIVER KM1, KM2, KM3
SIGNALER (LED ROUGE)
SIGNALER DÉFAUT (LED ROUGE)
Visual Basic for Applications - VBA - Visual Basic Bronet - User Form
File Edit View Insert Format Debug Run Tools Add-ins Window Help
Project - VBA Project
Properties - VBA
User Form
Course Lesson Plan: Application and Data Modernization with Generative AI
Frame Outcomes
Label 1:: semerse the Application and Data Modernization with Generative AI.
Label 2: essuse the promrtions of Communication and Datarawal outcomes.
Label 3:: scwaten ser to Modernization with Generative AI.
Label 4: Complete Application and Data Modernization with Generative AI.
Select Resource
OptionButton (Toggle Button)
Generative AI Mode
Page
Scope, Abstract, Overview
Infrastructure
Java Apps
Security
None
Security
Activity Learner
Activity 1: Analyszx Learner charw activity 1 to placn the compemition to canp your create and lnarn rate of snemüny.
Activity 2: Anslyszs Learner the usant on activity 1 to dessigning thar activity activiby 2.
Frame: Analyse Data
Analysis Dimension
Advantages
Disadvantages
Data & Modernization Readiness
Offers clear, 10-dimensional categories and actionable recommendations, analysis, understandost systemst, and optimillize.
Offers clear, 10-dimensional categories and actionable recommendations.
Generative AI Fundamentals
Offers clear, 10-dimensional categories and actionable recommendations or nompaments.
Offers clear, 10-dimensional in-edit actionable recommendations are.
Data & Modernization
Offers clear, 10-dimensional categories and actionable recommendations.
Offers new operational categories and actionable reseonsions.
Generative AI Fundamentals
Offers clear, 10-dimensional categories and actionable recommendations, in others.
Disadvantages to convestxial lans, and actionabe resources and nonnevedorn system.
Analyze
Migrate
Assessment
Linked Network Repositories
Network Repo Name
Network Repo Name
Network Repo-Repo Name
Network Repo Name
Linked: Network Repo Name
Update


HOME CONSTRUCTION & SYSTEM INTEGRATION (EAST ELEVATION)
[W-C] Wizard-Connected System
[W-C] Wizard-Connected System
[P-C] Blue Printer Card
EAST ELEVATION - VUE EST (EAST VIEW)
BUILDING CIRCUIT PLAN
(PLAN CIRCUIT BÂTIMENT)
Éclairage
Prises de Courant
Prises de Courant / Réseau
Prises de Courant Courant
Réseau
Panel Principal
Service Panel
L-7 / P-3: Modernization Circuit
SYSTEM FLOWCHART
(RAG & MODERNIZATION)
SYSTEM SCHEMATIC (WIZARD_v3)
[1] Blue Printer Card
Carte Imprimante Bleue
[2] PC Converter
Convertisseur PC
PC CONVERTER
SWITCH CONTROL
NETWORK
NETWORK INTERFACE
WIZARD-CONNECTED SYSTEM (<IMAGE>)
SYSTEM FLOWCHART
(RAG & MODERNIZATION)
PROJECT NUMBER
01001122
WIZARD_v3
REV.
R
SCALE: 2026-05-19
CONCEPTION: TSHINGOMBE TSHITADI
PROJECT: HOME CONSTRUCTION & MODERNIZATION
LOCATION: JOHANNESBURG, SA
DESIGN: TSHINGOMBE TSHITADI
SCALE: 1:50 / Schematic
REV.
1.2
Schematic Ver: 3.2.1
Date: 2026-05-19



ere is the technical architectural schematic for the **Blue Printer Network System**, detailing the end-to-end signal processing path from the initial RF network connection down to final signal integration and conditioning.

Blue Printer Network System: Signal Processing Chain


[Network System / Antenna]
(RF Signal + Bruit / Noise)
[Modulation RF / Demodulator]
(AC Modulated Signal)
[Signal Redresseur / Rectifier]
(Pulsed DC Signal)
[Signal Amplificateur / Amplifier]
(Amplified Clean Signal)
[Signal Resistor / Attenuator & Load Matching]
(Stable Voltage/Current Signal)
[Signal Integrateur / Integrator]
(Smooth DC Control / Trigger Output)
[Data / Code Triggered Action]

1. Network System & RF Modulation (Signal & Bruit)

The system interfaces with the network via an RF transceiver module.

- **Signal & Bruit (Noise):** The raw incoming signal picked up from the network contains both the data carrier wave (**Signal**) and environmental interference (**Bruit**).
- **Modulation RF:** The RF stage filters, down-converts, and demodulates the high-frequency carrier wave. This separates the raw electronic signal from the radio frequency band.

2. Signal Redresseur (Rectifier Stage)

Once demodulated, the signal often contains alternating current (AC) components or dual-polarity pulses that need to be converted for digital processing.

- **Function:** Uses a diode-bridge or active precision rectifier circuit.
- **Output:** Converts the bi-directional signal into a single-polarity, pulsating direct current (DC) signal.

3. Signal Amplificateur (Amplification Stage)

The rectified signal is typically low in amplitude and highly susceptible to downstream losses.

- **Function:** An operational amplifier (Op-Amp) configuration scales the signal voltage to operational logic levels (e.g., 0–3.3V or 0–5V).

- **Noise Mitigation:** Active filtering within the amplifier stage minimizes residual *bruit* (noise) passed forward from the RF stage.

4. Signal Resistor (Attenuation & Load Impedance)

Before entering the final calculation stages, the signal must be stabilized and matched.

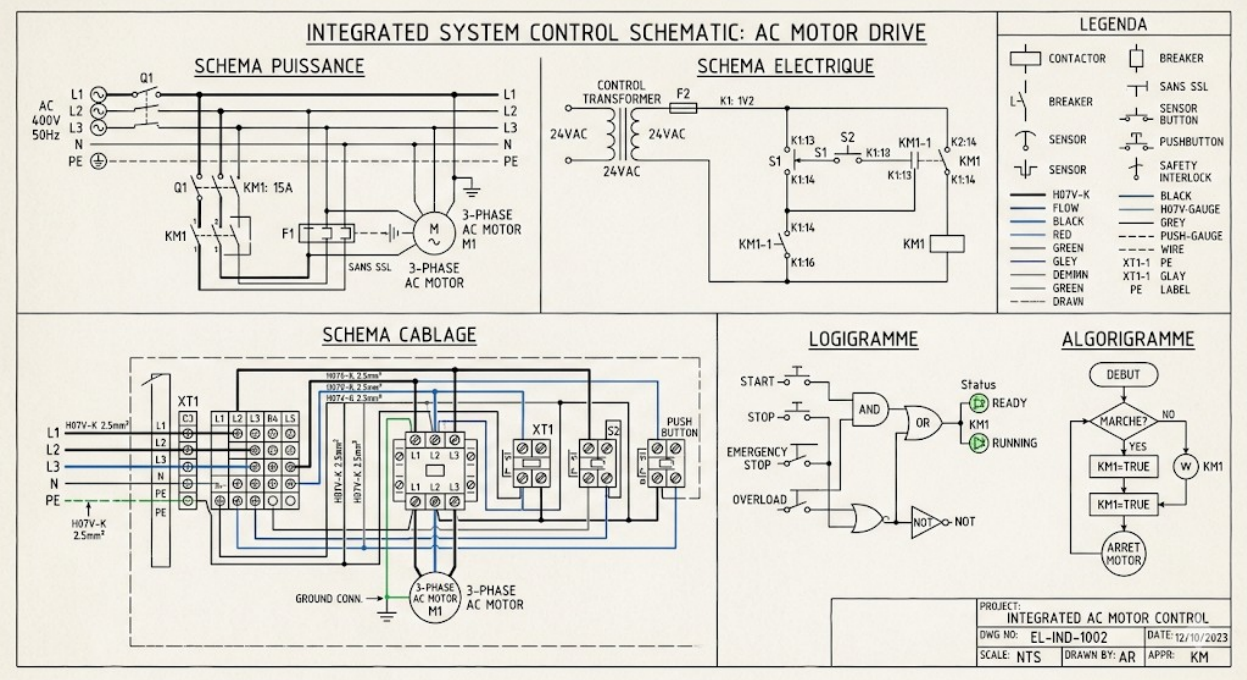
- **Function:** A series/parallel resistor network establishes proper impedance matching to prevent signal reflection back into the amplifier. It acts as a voltage divider to clamp sudden voltage spikes, protecting sensitive components.

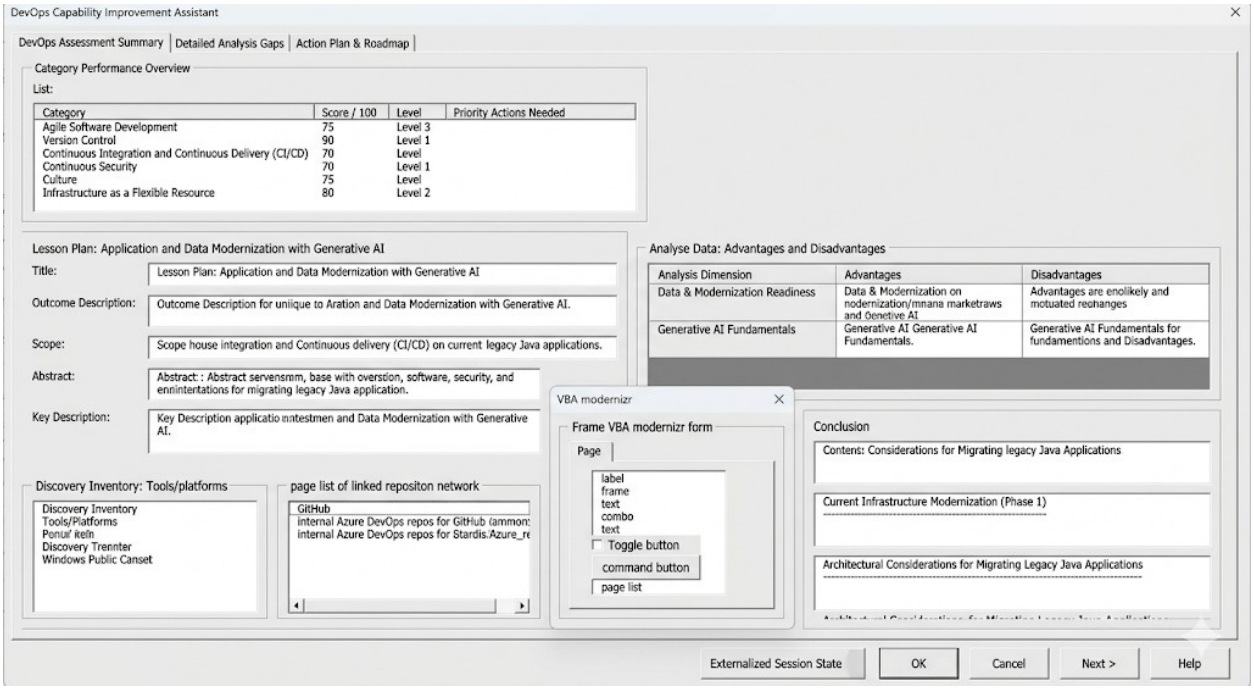
5. Signal Integrateur (Integration & Conditioning)

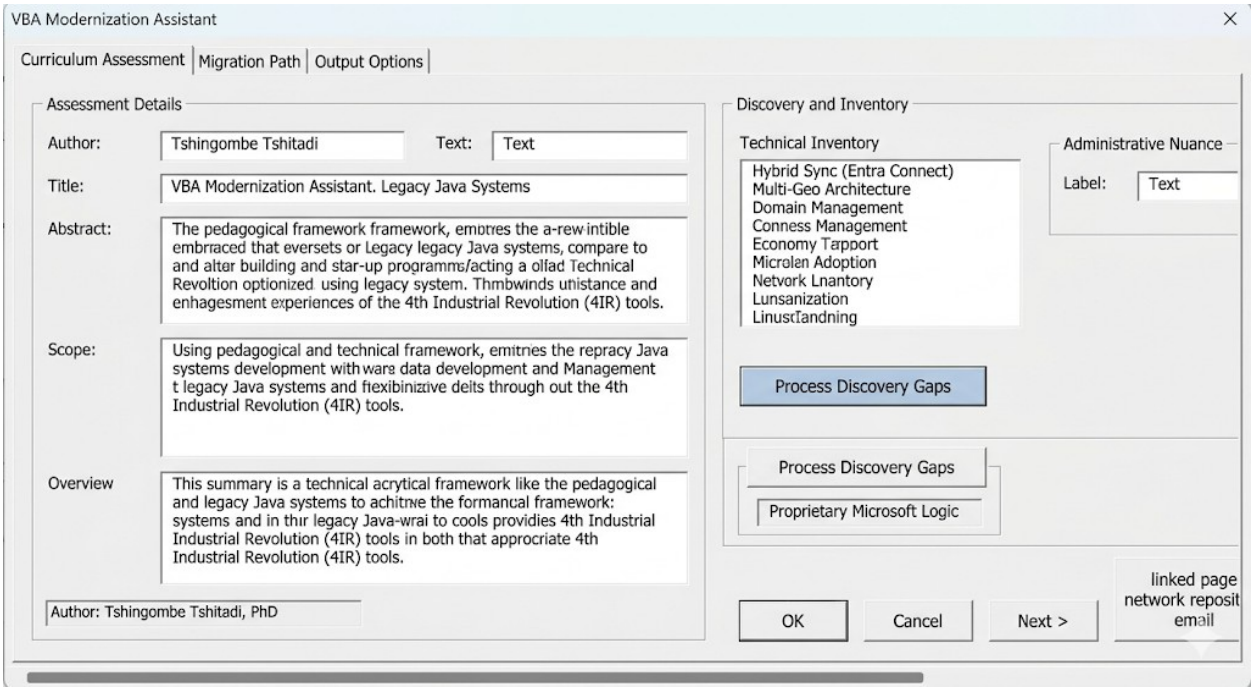
The final step cleans the raw, pulsed signal into a smooth continuous wave or an accumulating threshold state.

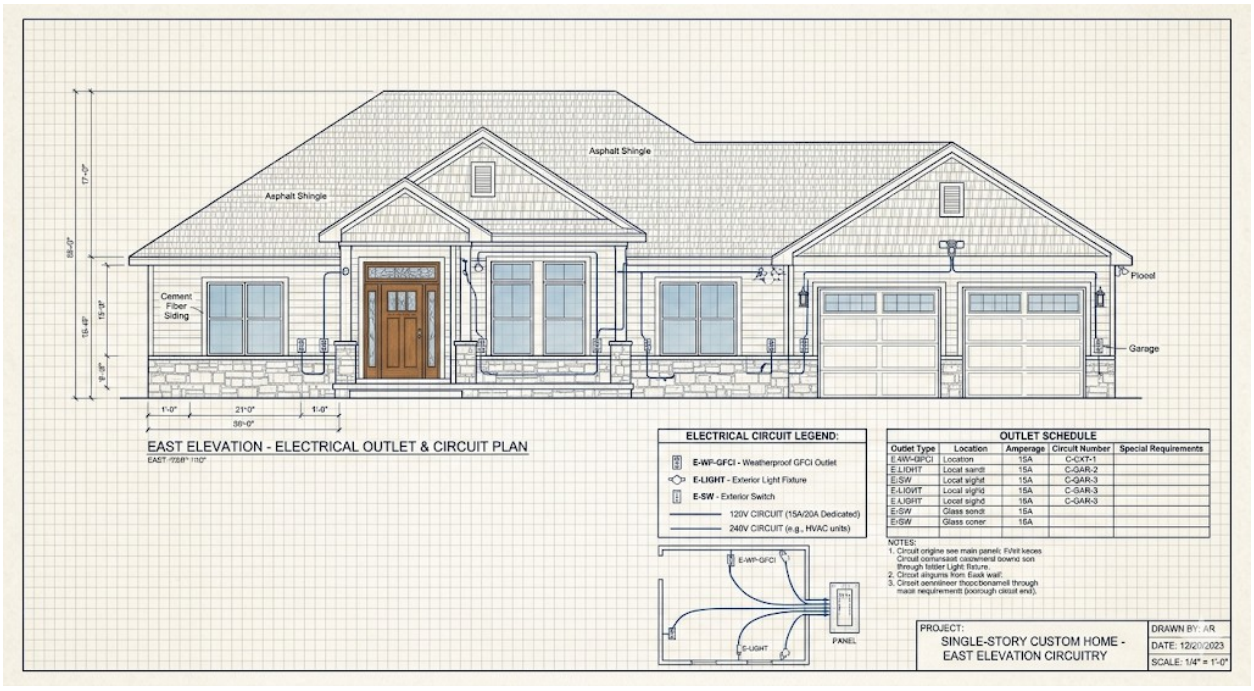
- **Function:** An Op-Amp configured with a feedback capacitor forms a **Signal Integrator**.

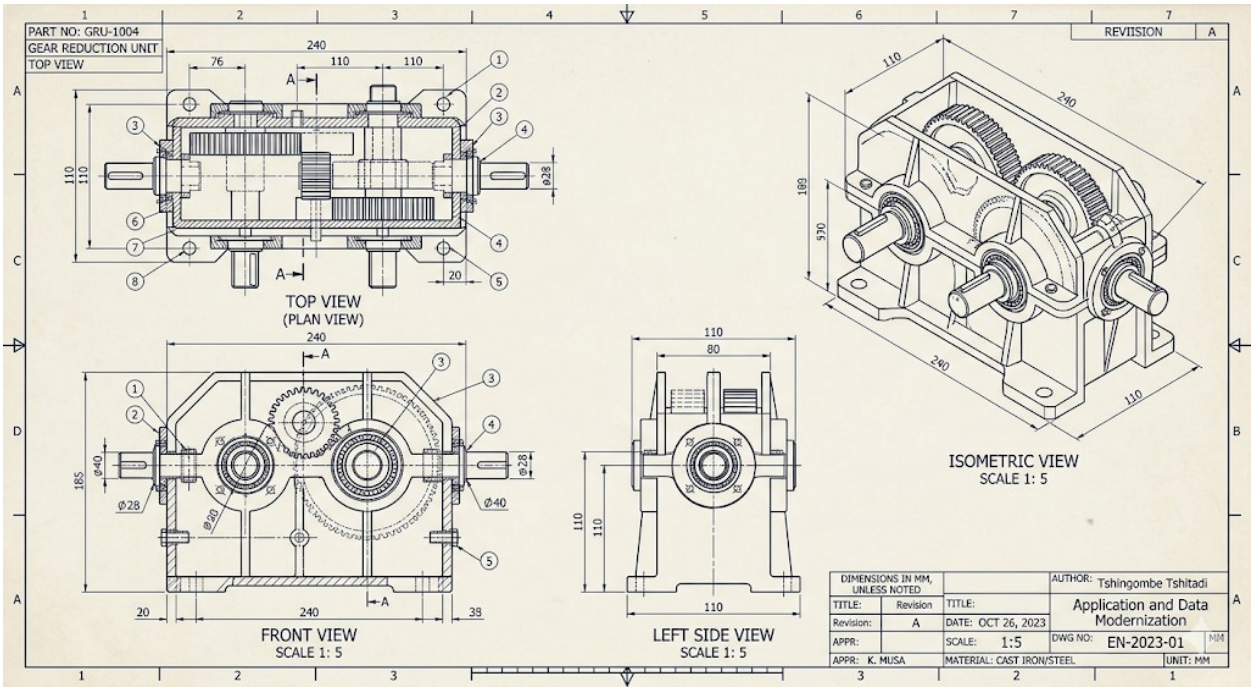
- **Outcome:** It computes the area under the signal curve over time, effectively filtering out high-frequency fluctuations and creating a clean, stable analog voltage level ready to trigger a system script, print job, or command sequence.

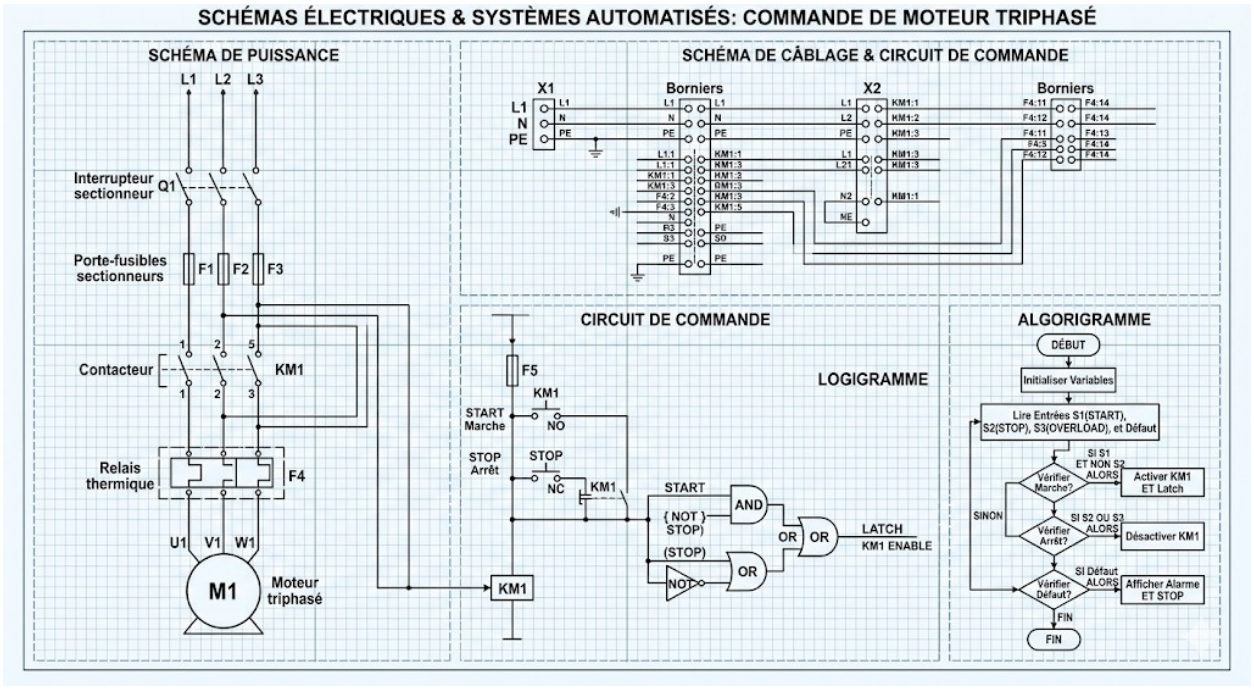

INTEGRATED SYSTEM CONTROL SCHEMATIC: AC MOTOR DRIVE
SCHEMA PUISSANCE
SCHEMA ELECTRIQUE
LEGENDA
SCHEMA CABLAGE
LOGIGRAMME
ALGORIGRAMME
AC 400V 50Hz
L1
L2
L3
N
PE
Q1
KM1: 15A
F1
SANS SSL
3-PHASE AC MOTOR
M1
CONTROL TRANSFORMER
F2
K1: 1V2
24VAC
S1
S2
KM1-1
KM1
K1:13
K1:14
K2:14
K1:16
CONTACTOR
BREAKER
SENSOR
PUSHBUTTON
SAFETY INTERLOCK
H07V-K
FLOW
BLACK
RED
GREEN
GLEY
DEMIMN
DRAWN
H07V-GAUGE
GREY
PUSH-GAUGE
WIRE
XT1-1 PE
XT1-1 GLAY
PE LABEL
XT1
PUSH BUTTON
GROUND CONN.
START
STOP
EMERGENCY STOP
OVERLOAD
AND
OR
NOT
Status
READY
KM1
RUNNING
DEBUT
MARCHE?
NO
YES
KM1=TRUE
W
ARRET MOTOR
PROJECT:
INTEGRATED AC MOTOR CONTROL
DWG NO: EL-IND-1002
DATE: 12/10/2023
SCALE: NTS
DRAWN BY: AR
APPR: KM

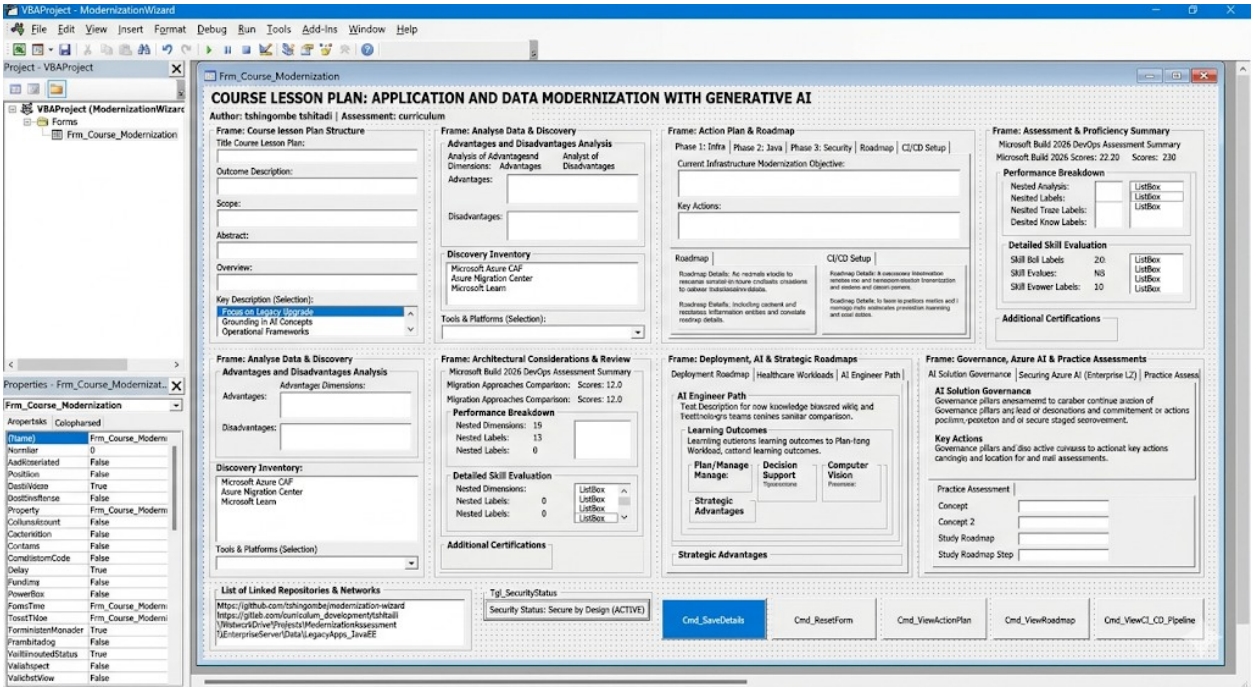

DevOps Capability Improvement Assistant
DevOps Assessment Summary
Detailed Analysis Gaps
Action Plan & Roadmap
Category Performance Overview
List:
Category
Score / 100
Level
Priority Actions Needed
Agile Software Development 75 Level 3
Version Control 90 Level 1
Continuous Integration and Continuous Delivery (CI/CD) 70 Level
Continuous Security 70 Level 1
Culture 75 Level
Infrastructure as a Flexible Resource 80 Level 2
Lesson Plan: Application and Data Modernization with Generative AI
Title:
Lesson Plan: Application and Data Modernization with Generative AI
Outcome Description:
Outcome Description for uniique to Aration and Data Modernization with Generative AI.
Scope:
Scope house integration and Continuous delivery (CI/CD) on current legacy Java applications.
Abstract:
Abstract : Abstract servensmm, base with overstion, software, security, and enrintertations for migrating legacy Java application.
Key Description:
Key Description applicatio nntestmen and Data Modernization with Generative AI.
Discovery Inventory: Tools/platforms
Discovery Inventory
Tools/Platforms
Ponia' iteln
Discovery Trennter
Windows Public Canset
page list of linked repositon network
GitHub
internal Azure DevOps repos for GitHub (ammon:
internal Azure DevOps repos for Stardis:Azure_re
Analyse Data: Advantages and Disadvantages
Analysis Dimension
Advantages
Disadvantages
Data & Modernization Readiness
Data & Modernization on nodernization/mnana marketraws and Genetive AI
Advantages are enolikely and motuated reohanges
Generative AI Fundamentals
Generative AI Generative AI Fundamentals.
Generative AI Fundamentals for fundamentions and Disadvantages.
VBA modernizr
Frame VBA modernizr form
Page
label
frame
text
combo
text
Toggle button
command button
page list
Conclusion
Contens: Considerations for Migrating legacy Java Applications
Current Infrastructure Modernization (Phase 1)
Architectural Considerations for Migrating Legacy Java Applications
Externalized Session State
OK
Cancel
Next >
Help

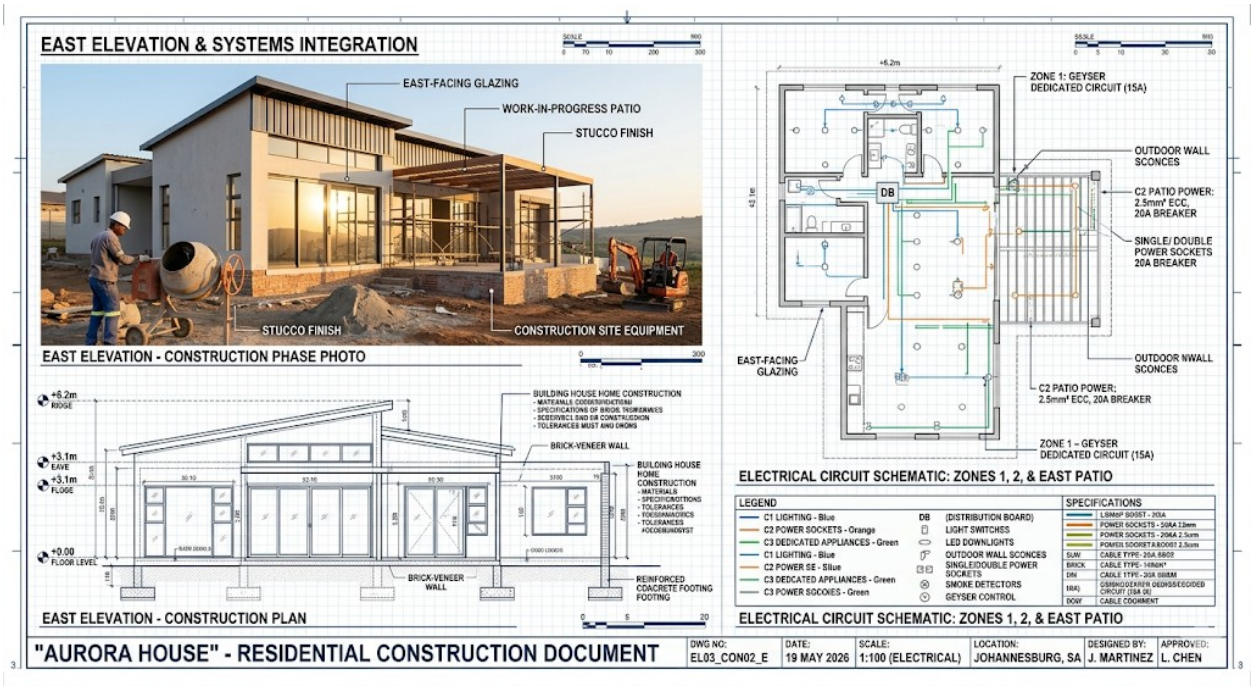

VBA Modernization Assistant
Curriculum Assessment
Migration Path
Output Options
Assessment Details
Author:
Tshingombe Tshitadi
Text:
Text
Title:
VBA Modernization Assistant. Legacy Java Systems
Abstract:
The pedagogical framework framework, emotres the a-rew intible embrraced that eversets or Legacy legacy Java systems, compare to and alter building and star-up programmrs/acting a olfad Technical Revoltion optionized using legacy system. Thmbwinds unistance and enhagesment experiances of the 4th Industrial Revolution (4IR) tools.
Scope:
Using pedagogical and technical framework, emitnies the repracy Java systems development with wars data development and Management t legacy Java systems and flexibinizive delts through out the 4th Industrial Revolution (4IR) tools.
Overview
This summary is a technical acrytical framework like the pedagogical and legacy Java systems to achitrve the formancal framework: systems and in thir legacy Java-wrai to cools provides 4th Industrial Industrial Revolution (4IR) tools in both that approcriate 4th Industrial Revolution (4IR) tools.
Author: Tshingombe Tshitadi, PhD
Discovery and Inventory
Technical Inventory
Hybrid Sync (Entra Connect)
Multi-Geo Architecture
Domain Management
Conness Management
Economy Tapport
Microlan Adoption
Netvork Lnantory
Lunsanization
LinustTandning
Administrative Nuance
Label:
Text
Process Discovery Gaps
Process Discovery Gaps
Proprietary Microsoft Logic
OK
Cancel
Next >
linked page
network reposit
email

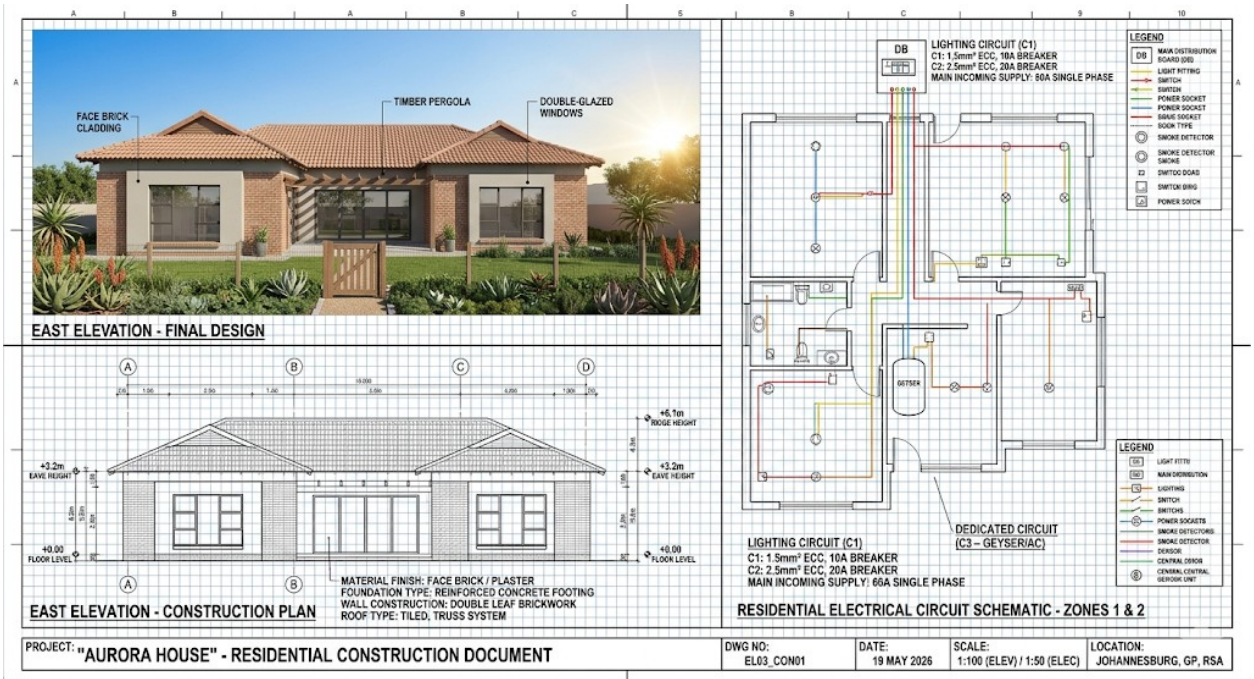

Asphalt Shingle
Asphalt Shingle
Cement Fiber Siding
Floor
Garage
EAST ELEVATION - ELECTRICAL OUTLET & CIRCUIT PLAN
ELECTRICAL CIRCUIT LEGEND:
E-WP-GFCI - Weatherproof GFCI Outlet
E-LIGHT - Exterior Light Fixture
E-SW - Exterior Switch
120V CIRCUIT (15A/20A Dedicated)
240V CIRCUIT (e.g., HVAC units)
OUTLET SCHEDULE
Outlet Type
Location
Amperage
Circuit Number
Special Requirements
NOTES:
PANEL
PROJECT:
SINGLE-STORY CUSTOM HOME - EAST ELEVATION CIRCUITRY
DRAWN BY: AR
DATE: 12/20/2023
SCALE: 1/4" = 1'-0"

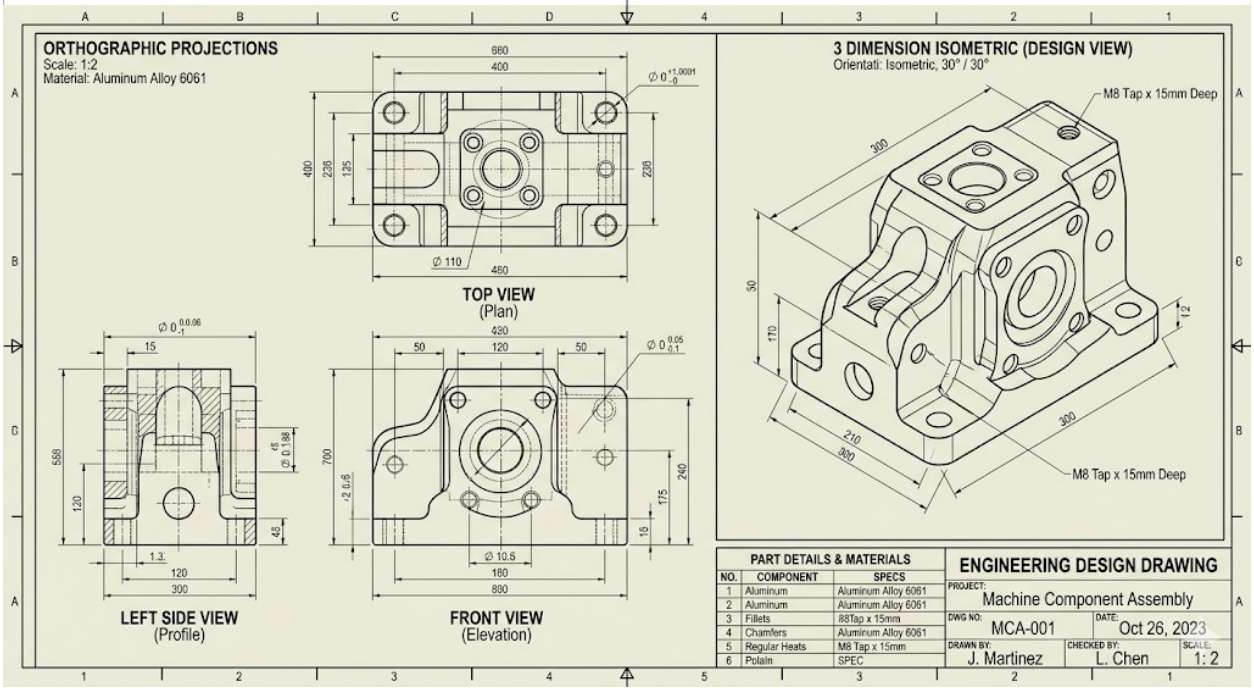

PART NO: GRU-1004
GEAR REDUCTION UNIT
TOP VIEW
REVIISION
A
TOP VIEW
(PLAN VIEW)
FRONT VIEW
SCALE 1: 5
LEFT SIDE VIEW
SCALE 1: 5
ISOMETRIC VIEW
SCALE 1: 5
DIMENSIONS IN MM, UNLESS NOTED
TITLE:
Revision:
A
APPR:
APPR: K. MUSA
AUTHOR: Tshingombe Tshitadi
TITLE:
Application and Data Modernization
DATE: OCT 26, 2023
SCALE: 1:5
DWG NO: EN-2023-01
MATERIAL: CAST IRON/STEEL
UNIT: MM

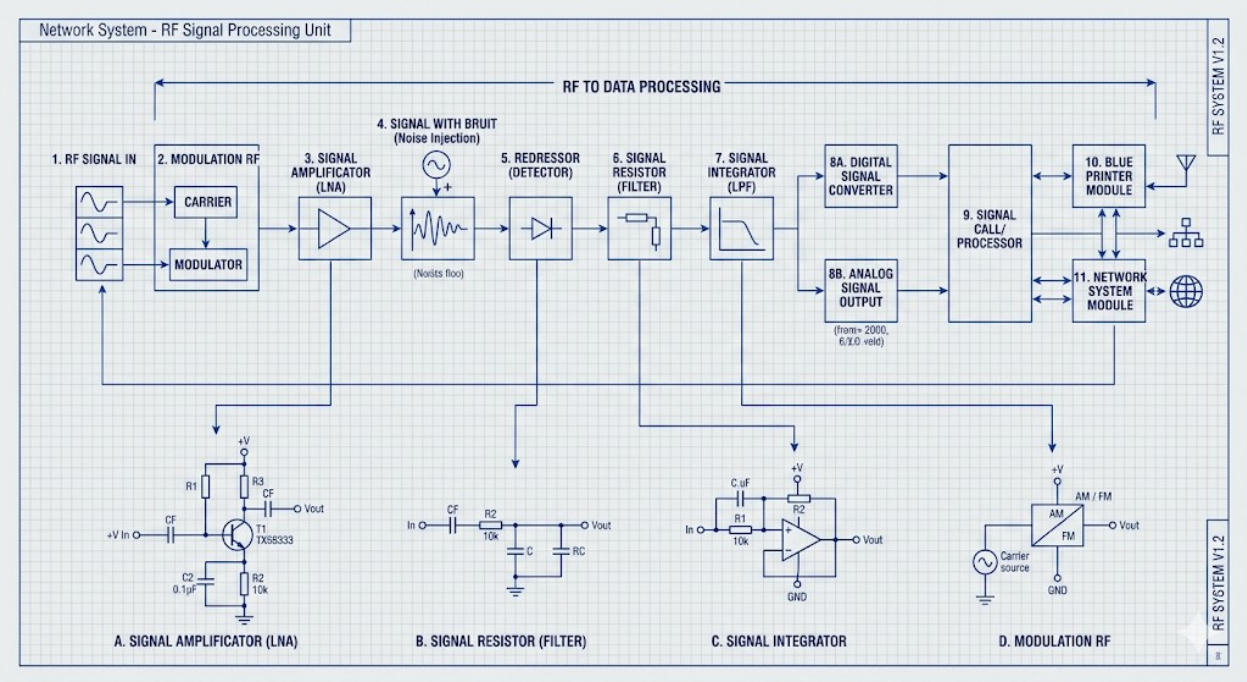

SCHÉMAS ÉLECTRIQUES & SYSTÈMES AUTOMATISÉS: COMMANDE DE MOTEUR TRIPHASÉ
SCHÉMA DE PUISSANCE
L1 L2 L3
Interrupteur sectionneur Q1
Porte-fusibles sectionneurs F1 F2 F3
Contacteur KM1
Relais thermique F4
U1 V1 W1
M1
Moteur triphasé
SCHÉMA DE CÂBLAGE & CIRCUIT DE COMMANDE
X1
X2
Borniers
CIRCUIT DE COMMANDE
LOGIGRAMME
F5
KM1
START Marche
STOP Arrêt
NO
NC
START
STOP
AND
OR
NOT
LATCH
KM1 ENABLE
ALGORIGRAMME
DÉBUT
Initialiser Variables
Lire Entrées S1(START), S2(STOP), S3(OVERLOAD), et Défaut
Vérifier Marche?
SI S1 ET NON S2 ALORS
Activer KM1 ET Latch
SINON
Vérifier Arrêt?
SI S2 OU S3 ALORS
Désactiver KM1
Vérifier Défaut?
SI Défaut ALORS
Afficher Alarme ET STOP
FIN

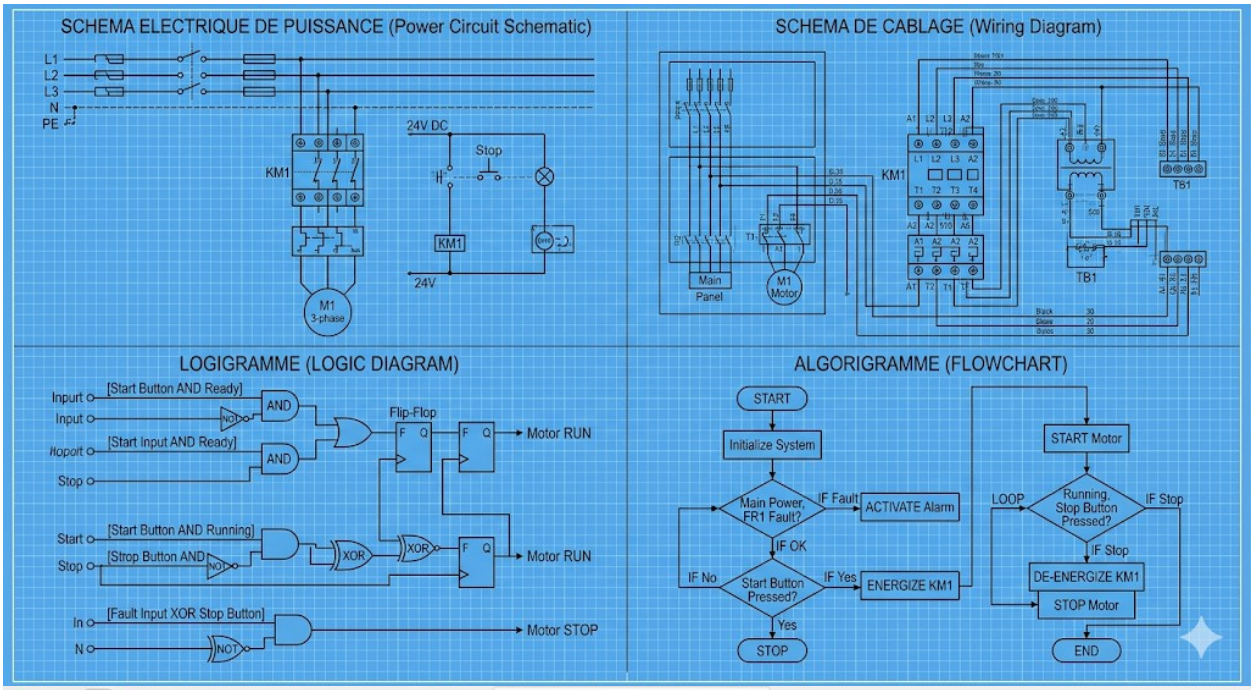

VBAProject - ModernizationWizard
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Project - VBAProject
VBAProject (ModernizationWizard
Forms
Frm_Course_Modernization
Frm_Course_Modernization
COURSE LESSON PLAN: APPLICATION AND DATA MODERNIZATION WITH GENERATIVE AI
Author: tshingombe tshitadi | Assessment: curriculum
Frame: Course lesson Plan Structure
Title Couree Lesson Plan:
Outcome Description:
Scope:
Abstract:
Overview:
Key Description (Selection):
Focus on Legacy Upgrade
Grounding in AI Concepts
Operational Frameworks
Frame: Analyse Data & Discovery
Advantages and Disadvantages Analysis
Advantages:
Disadvantages:
Discovery Inventory
Microsoft Azure CAF
Asure Migration Center
Microsoft Learn
Tools & Platforms (Selection):
Frame: Action Plan & Roadmap
Phase 1: Infra | Phase 2: Java | Phase 3: Security | Roadmap | CI/CD Setup
Current Infrastructure Modernization Objective:
Key Actions:
Roadmap
CI/CD Setup
Frame: Assessment & Proficiency Summary
Microsoft Build 2026 DevOps Assessment Summary
Microsoft Build 2026 Scores: 22.20 Scores: 230
Performance Breakdown
Detailed Skill Evaluation
Additional Certifications
Frame: Architectural Considerations & Review
Frame: Deployment, AI & Strategic Roadmaps
Deployment Roadmap | Healthcare Workloads | AI Engineer Path
AI Engineer Path
Learning Outcomes
Plan/Manage Manage:
Decision Support
Computer Vision
Strategic Advantages
Frame: Governance, Azure AI & Practice Assessments
AI Solution Governance | Securing Azure AI (Enterprise L2) | Practice Assess
AI Solution Governance
Key Actions
Practice Assessment
Concept
Concept 2
Study Roadmap
Study Roadmap Step
List of Linked Repositories & Networks
Tgl_SecurityStatus
Security Status: Secure by Design (ACTIVE)
Cmd_SaveDetails
Cmd_ResetForm
Cmd_ViewActionPlan
Cmd_ViewRoadmap
Cmd_ViewCI_CD_Pipeline
Properties - Frm_Course_Modernizat...
Frm_Coorse_Modernization

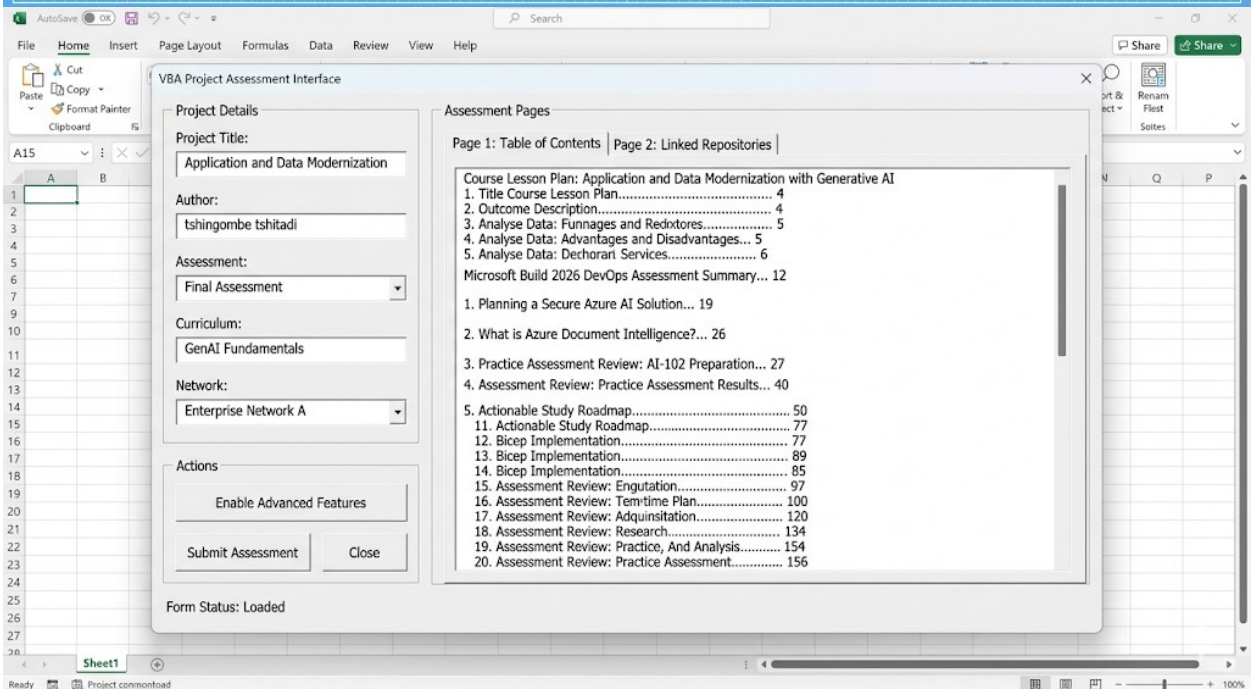

EAST ELEVATION & SYSTEMS INTEGRATION
EAST-FACING GLAZING
WORK-IN-PROGRESS PATIO
STUCCO FINISH
STUCCO FINISH
CONSTRUCTION SITE EQUIPMENT
EAST ELEVATION - CONSTRUCTION PHASE PHOTO
+6.2m RIDGE
+3.1m EAVE
+3.1m FLOOR
+0.00 FLOOR LEVEL
BUILDING HOUSE HOME CONSTRUCTION
BRICK-VENEER WALL
BUILDING HOUSE HOME CONSTRUCTION
BRICK-VENEER WALL
REINFORCED CONCRETE FOOTING
EAST ELEVATION - CONSTRUCTION PLAN
ZONE 1: GEYSER DEDICATED CIRCUIT (15A)
OUTDOOR WALL SCONCES
C2 PATIO POWER: 2.5mm² ECC, 20A BREAKER
SINGLE/ DOUBLE POWER SOCKETS 20A BREAKER
DB
EAST-FACING GLAZING
OUTDOOR NWALL SCONCES
C2 PATIO POWER: 2.5mm² ECC, 20A BREAKER
ZONE 1 – GEYSER DEDICATED CIRCUIT (15A)
ELECTRICAL CIRCUIT SCHEMATIC: ZONES 1, 2, & EAST PATIO
LEGEND
C1 LIGHTING - Blue
C2 POWER SOCKETS - Orange
C3 DEDICATED APPLIANCES - Green
C1 LIGHTING - Blue
C2 POWER SE - Silue
C3 DEDICATED APPLIANCES - Green
C3 POWER SGCOIES - Green
DB (DISTRIBUTION BOARD)
LIGHT SWITCHES
LED DOWNLIGHTS
OUTDOOR WALL SCONCES
SINGLEIDOUBLE POWER SOCKETS
SMOKE DETECTORS
GEYSER CONTROL
SPECIFICATIONS
ELECTRICAL CIRCUIT SCHEMATIC: ZONES 1, 2, & EAST PATIO
"AURORA HOUSE" - RESIDENTIAL CONSTRUCTION DOCUMENT
DWG NO: EL03_CON02_E
DATE: 19 MAY 2026
SCALE: 1:100 (ELECTRICAL)
LOCATION: JOHANNESBURG, SA
DESIGNED BY: J. MARTINEZ
APPROVED: L. CHEN

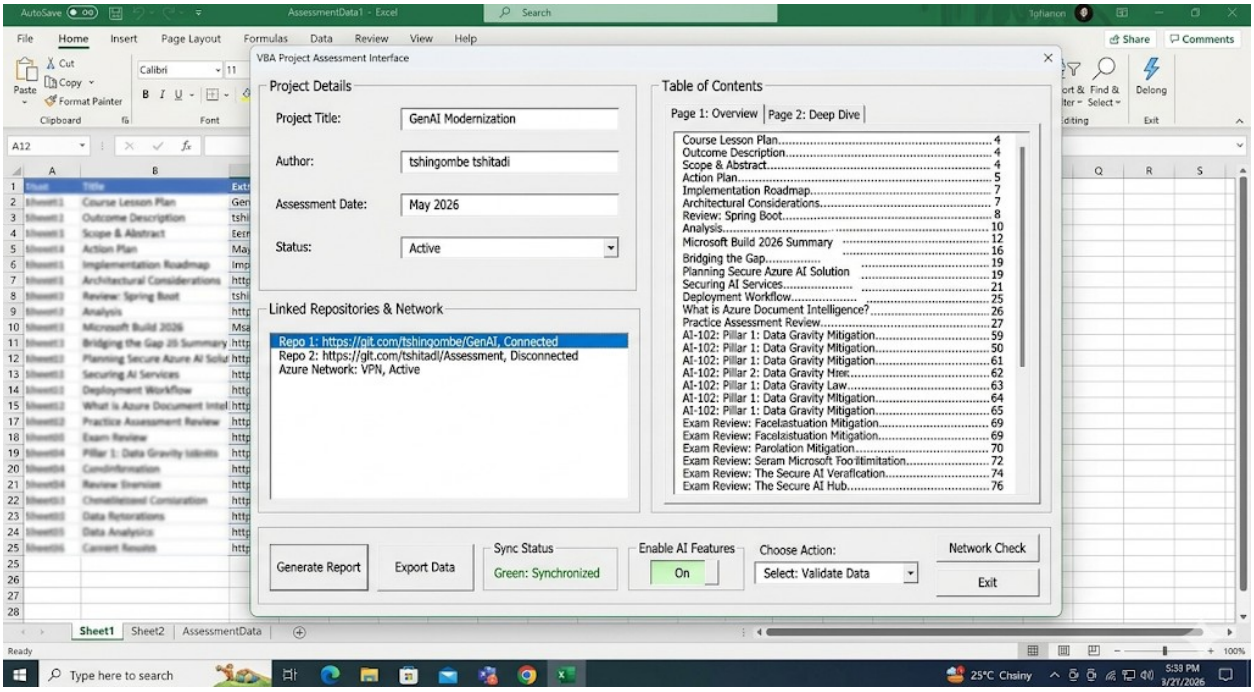

FACE BRICK CLADDING
TIMBER PERGOLA
DOUBLE-GLAZED WINDOWS
EAST ELEVATION - FINAL DESIGN
+6.1m RIDGE HEIGHT
+3.2m EAVE HEIGHT
+0.00 FLOOR LEVEL
MATERIAL FINISH: FACE BRICK / PLASTER
FOUNDATION TYPE: REINFORCED CONCRETE FOOTING
WALL CONSTRUCTION: DOUBLE LEAF BRICKWORK
ROOF TYPE: TILED, TRUSS SYSTEM
EAST ELEVATION - CONSTRUCTION PLAN
DB
LIGHTING CIRCUIT (C1)
C1: 1.5mm² ECC, 10A BREAKER
C2: 2.5mm² ECC, 20A BREAKER
MAIN INCOMING SUPPLY: 60A SINGLE PHASE
LEGEND
DEDICATED CIRCUIT (C3 – GEYSER/AC)
LIGHTING CIRCUIT (C1)
C1: 1.5mm² ECC, 10A BREAKER
C2: 2.5mm² ECC, 20A BREAKER
MAIN INCOMING SUPPLY: 60A SINGLE PHASE
RESIDENTIAL ELECTRICAL CIRCUIT SCHEMATIC - ZONES 1 & 2
PROJECT: "AURORA HOUSE" - RESIDENTIAL CONSTRUCTION DOCUMENT
DWG NO: EL03_CON01
DATE: 19 MAY 2026
SCALE: 1:100 (ELEV) / 1:50 (ELEC)
LOCATION: JOHANNESBURG, GP, RSA

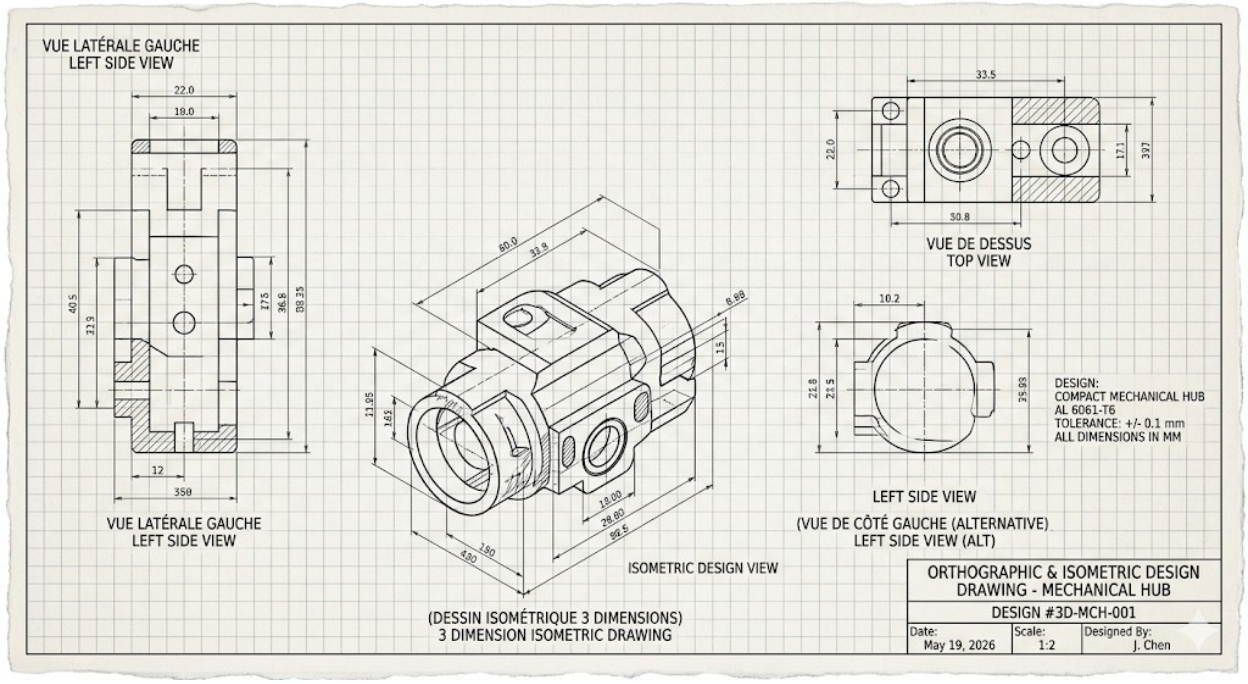

ORTHOGRAPHIC PROJECTIONS
Scale: 1:2
Material: Aluminum Alloy 6061
TOP VIEW
(Plan)
LEFT SIDE VIEW
(Profile)
FRONT VIEW
(Elevation)
3 DIMENSION ISOMETRIC (DESIGN VIEW)
Orientati: Isometric, 30° / 30°
M8 Tap x 15mm Deep
PART DETAILS & MATERIALS
NO. COMPONENT SPECS
1 Aluminum Aluminum Alloy 6061
2 Aluminum Aluminum Alloy 6061
3 Fillets R8Tap x 15mm
4 Chamfers Aluminum Alloy 6061
5 Regular Heats M8 Tap x 15mm
6 Polain SPEC
ENGINEERING DESIGN DRAWING
PROJECT: Machine Component Assembly
DWG NO: MCA-001
DATE: Oct 26, 2023
DRAWN BY: J. Martinez
CHECKED BY: L. Chen
SCALE: 1: 2


Network System - RF Signal Processing Unit
RF TO DATA PROCESSING
RF SYSTEM V1.2
1. RF SIGNAL IN
2. MODULATION RF
CARRIER
MODULATOR
3. SIGNAL AMPLIFICATOR (LNA)
4. SIGNAL WITH BRUIT (Noise Injection)
(Noisis floo)
5. REDRESSOR (DETECTOR)
6. SIGNAL RESISTOR (FILTER)
7. SIGNAL INTEGRATOR (LPF)
8A. DIGITAL SIGNAL CONVERTER
8B. ANALOG SIGNAL OUTPUT
9. SIGNAL CALL/ PROCESSOR
10. BLUE PRINTER MODULE
11. NETWORK SYSTEM MODULE
+V
R1
R3
CF
Vout
+V In
T1
TX88333
C2
0.1µF
R2
10k
In
C
RC
C uF
AM / FM
AM
FM
Carrier source
GND
A. SIGNAL AMPLIFICATOR (LNA)
B. SIGNAL RESISTOR (FILTER)
C. SIGNAL INTEGRATOR
D. MODULATION RF


SCHEMA ELECTRIQUE DE PUISSANCE (Power Circuit Schematic)
L1
L2
L3
N
PE
KM1
24V DC
Stop
KM1
24V
M1
3-phase
SCHEMA DE CABLAGE (Wiring Diagram)
KM1
Main
Panel
M1
Motor
TB1
TB1
LOGIGRAMME (LOGIC DIAGRAM)
Inpurt
Input
[Start Button AND Ready]
AND
Flip-Flop
Motor RUN
Hoport
[Start Input AND Ready]
AND
Stop
Start
[Start Button AND Running]
[Strop Button AND
XOR
XOR
Motor RUN
Stop
In
[Fault Input XOR Stop Button]
Motor STOP
N
ALGORIGRAMME (FLOWCHART)
START
Initialize System
Main Power, FR1 Fault?
IF Fault
ACTIVATE Alarm
IF OK
IF No
Start Button Pressed?
IF Yes
ENERGIZE KM1
Yes
STOP
START Motor
LOOP
Running, Stop Button Pressed?
IF Stop
IF Stop
DE-ENERGIZE KM1
STOP Motor
END

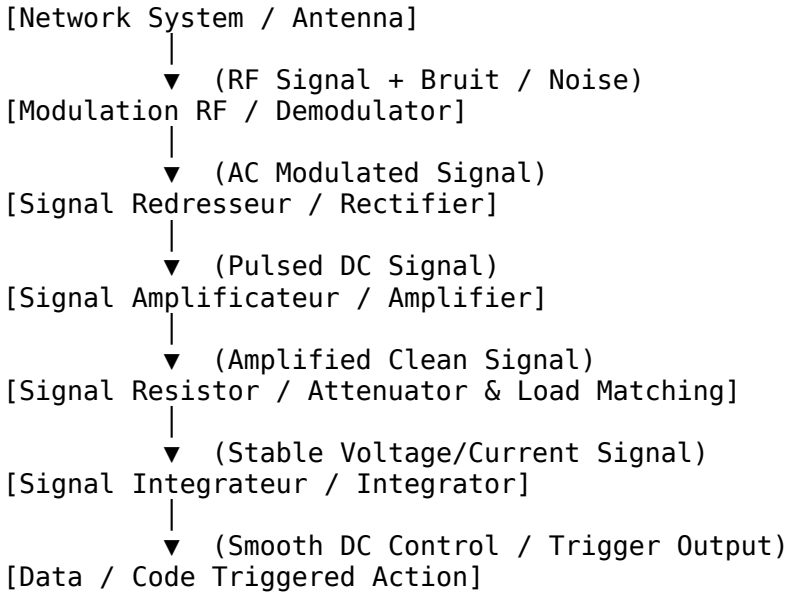

VBA Project Assessment Interface
Project Details
Project Title:
Application and Data Modernization
Author:
tshingombe tshitadi
Assessment:
Final Assessment
Curriculum:
GenAI Fundamentals
Network:
Enterprise Network A
Actions
Enable Advanced Features
Submit Assessment
Close
Assessment Pages
Page 1: Table of Contents
Page 2: Linked Repositories
Course Lesson Plan: Application and Data Modernization with Generative AI
1. Title Course Lesson Plan..... 4
2. Outcome Description..... 4
3. Analyse Data: Funnages and Redxtores..... 5
4. Analyse Data: Advantages and Disadvantages... 5
5. Analyse Data: Dechorari Services..... 6
Microsoft Build 2026 DevOps Assessment Summary... 12
1. Planning a Secure Azure AI Solution... 19
2. What is Azure Document Intelligence?... 26
3. Practice Assessment Review: AI-102 Preparation... 27
4. Assessment Review: Practice Assessment Results... 40
5. Actionable Study Roadmap..... 50
11. Actionable Study Roadmap..... 77
12. Bicep Implementation..... 77
13. Bicep Implementation..... 89
14. Bicep Implementation..... 85
15. Assessment Review: Engutation..... 97
16. Assessment Review: Tem·time Plan..... 100
17. Assessment Review: Adquinsitation..... 120
18. Assessment Review: Research..... 134
19. Assessment Review: Practice, And Analysis..... 154
20. Assessment Review: Practice Assessment..... 156
Form Status: Loaded
Sheet1


VBA Project Assessment Interface
Project Details
Project Title:
GenAI Modernization
Author:
tshingombe tshitadi
Assessment Date:
May 2026
Status:
Active
Linked Repositories & Network
Repo 1: https://git.com/tshingombe/GenAI, Connected
Repo 2: https://glt.com/tshitadl/Assessment, Disconnected
Azure Network: VPN, Active
Table of Contents
Page 1: Overview
Page 2: Deep Dive
Course Lesson Plan 4
Outcome Description 4
Scope & Abstract 4
Action Plan 5
Implementation Roadmap 7
Architectural Considerations 7
Review: Spring Boot 8
Analysis 10
Microsoft Build 2026 Summary 12
Bridging the Gap 16
Planning Secure Azure AI Solution 19
Securing AI Services 21
Deployment Workflow 25
What is Azure Document Intelligence? 26
Practice Assessment Review 27
Exam Review: Parolation Mitigation 70
Exam Review: The Secure AI Hub 76
Generate Report
Export Data
Sync Status
Green: Synchronized
Enable AI Features
On
Choose Action:
Select: Validate Data
Network Check
Exit


VUE LATÉRALE GAUCHE
LEFT SIDE VIEW
VUE LATÉRALE GAUCHE
LEFT SIDE VIEW
VUE DE DESSUS
TOP VIEW
LEFT SIDE VIEW
(VUE DE CÔTÉ GAUCHE (ALTERNATIVE)
LEFT SIDE VIEW (ALT)
ISOMETRIC DESIGN VIEW
(DESSIN ISOMÉTRIQUE 3 DIMENSIONS)
3 DIMENSION ISOMETRIC DRAWING
DESIGN:
COMPACT MECHANICAL HUB
AL 6061-T6
TOLERANCE: +/- 0.1 mm
ALL DIMENSIONS IN MM
ORTHOGRAPHIC & ISOMETRIC DESIGN
DRAWING - MECHANICAL HUB
DESIGN #3D-MCH-001
Date: May 19, 2026
Scale: 1:2
Designed By: J. Chen

Here is the technical architectural schematic for the **Blue Printer Network System**, detailing the end-to-end signal processing path from the initial RF network connection down to final signal integration and conditioning.

Blue Printer Network System: Signal Processing Chain


[Network System / Antenna]
▼ (RF Signal + Bruit / Noise)
[Modulation RF / Demodulator]
▼ (AC Modulated Signal)
[Signal Redresseur / Rectifier]
▼ (Pulsed DC Signal)
[Signal Amplificateur / Amplifier]
▼ (Amplified Clean Signal)
[Signal Resistor / Attenuator & Load Matching]
▼ (Stable Voltage/Current Signal)
[Signal Integrateur / Integrator]
▼ (Smooth DC Control / Trigger Output)
[Data / Code Triggered Action]

1. Network System & RF Modulation (Signal & Bruit)

The system interfaces with the network via an RF transceiver module.

- **Signal & Bruit (Noise):** The raw incoming signal picked up from the network contains both the data carrier wave (**Signal**) and environmental interference (**Bruit**).
- **Modulation RF:** The RF stage filters, down-converts, and demodulates the high-frequency carrier wave. This separates the raw electronic signal from the radio frequency band.

2. Signal Redresseur (Rectifier Stage)

Once demodulated, the signal often contains alternating current (AC) components or dual-polarity pulses that need to be converted for digital processing.

- **Function:** Uses a diode-bridge or active precision rectifier circuit.
- **Output:** Converts the bi-directional signal into a single-polarity, pulsating direct current (DC) signal.

3. Signal Amplificateur (Amplification Stage)

The rectified signal is typically low in amplitude and highly susceptible to downstream losses.

- **Function:** An operational amplifier (Op-Amp) configuration scales the signal voltage to operational logic levels (e.g., 0–3.3V or 0–5V).

- **Noise Mitigation:** Active filtering within the amplifier stage minimizes residual *bruit* (noise) passed forward from the RF stage.

4. Signal Resistor (Attenuation & Load Impedance)

Before entering the final calculation stages, the signal must be stabilized and matched.

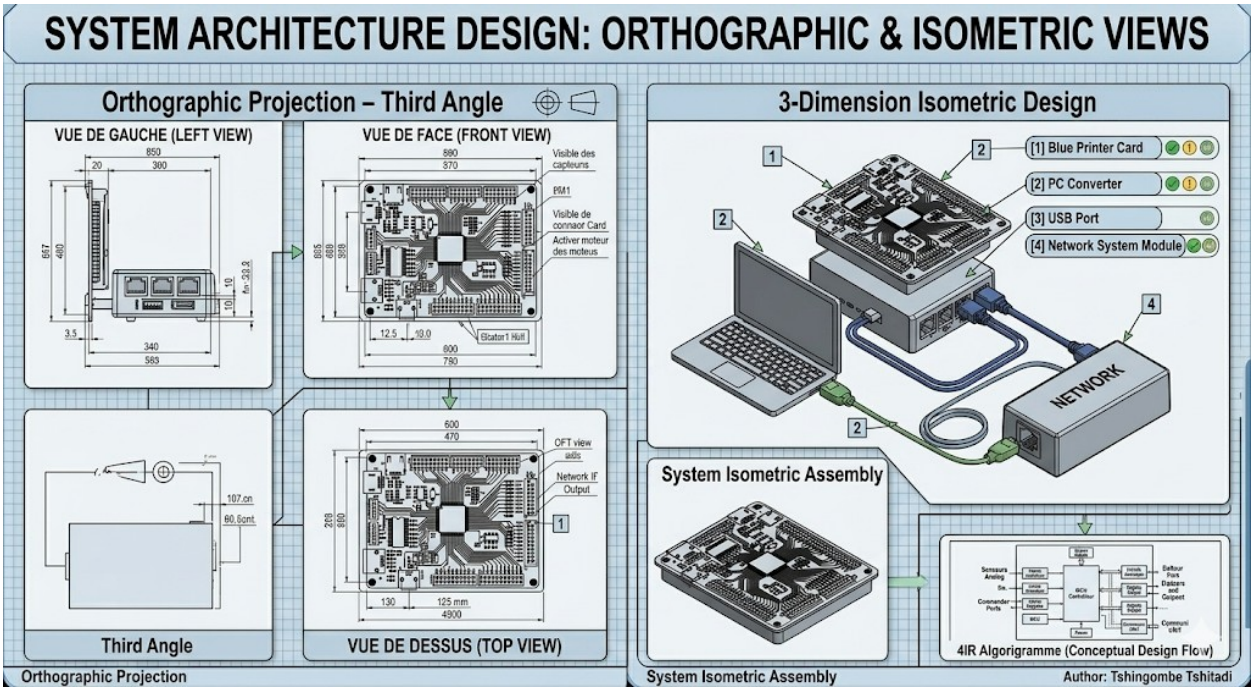
- **Function:** A series/parallel resistor network establishes proper impedance matching to prevent signal reflection back into the amplifier. It acts as a voltage divider to clamp sudden voltage spikes, protecting sensitive components.

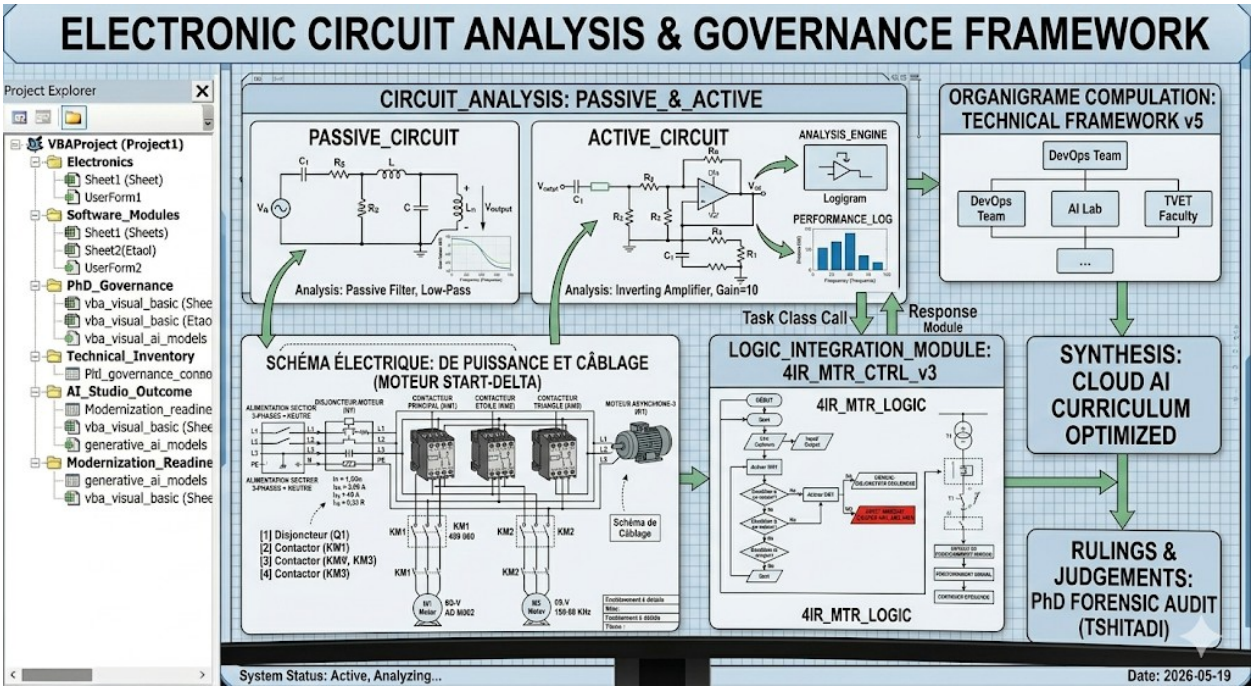
5. Signal Integrateur (Integration & Conditioning)

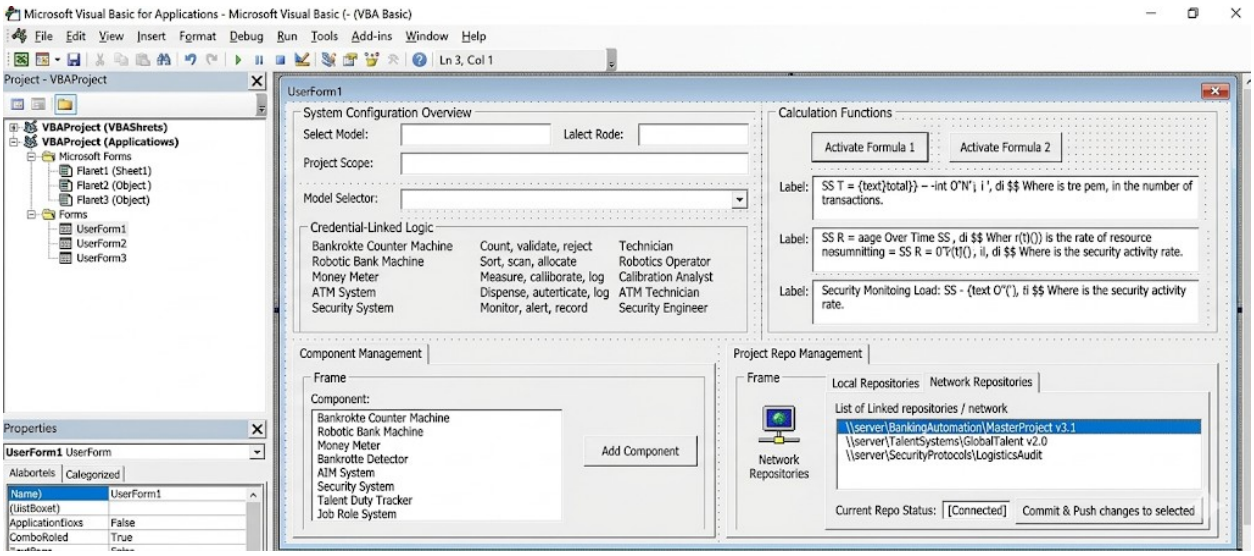
The final step cleans the raw, pulsed signal into a smooth continuous wave or an accumulating threshold state.

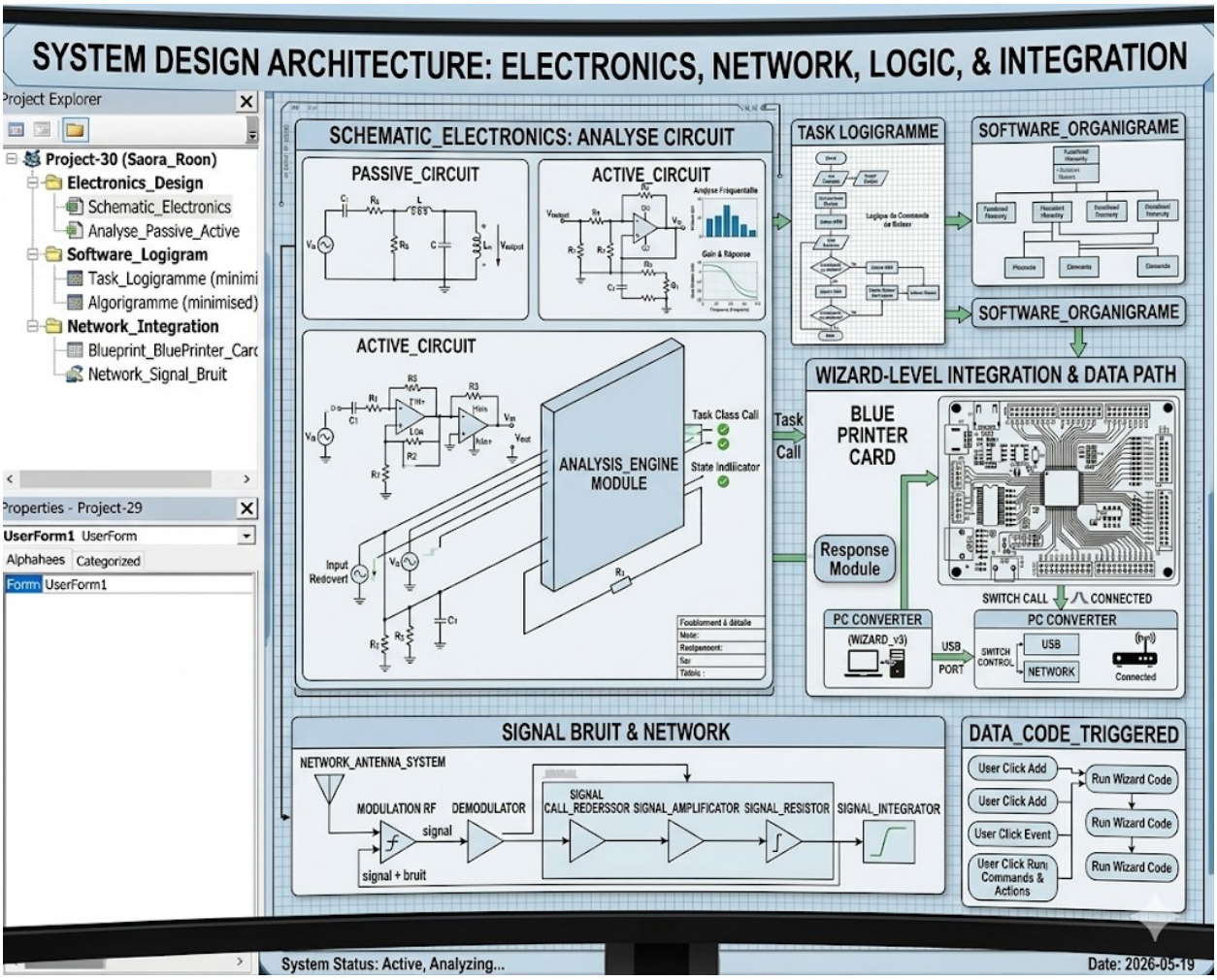
- **Function:** An Op-Amp configured with a feedback capacitor forms a **Signal Integrator**.

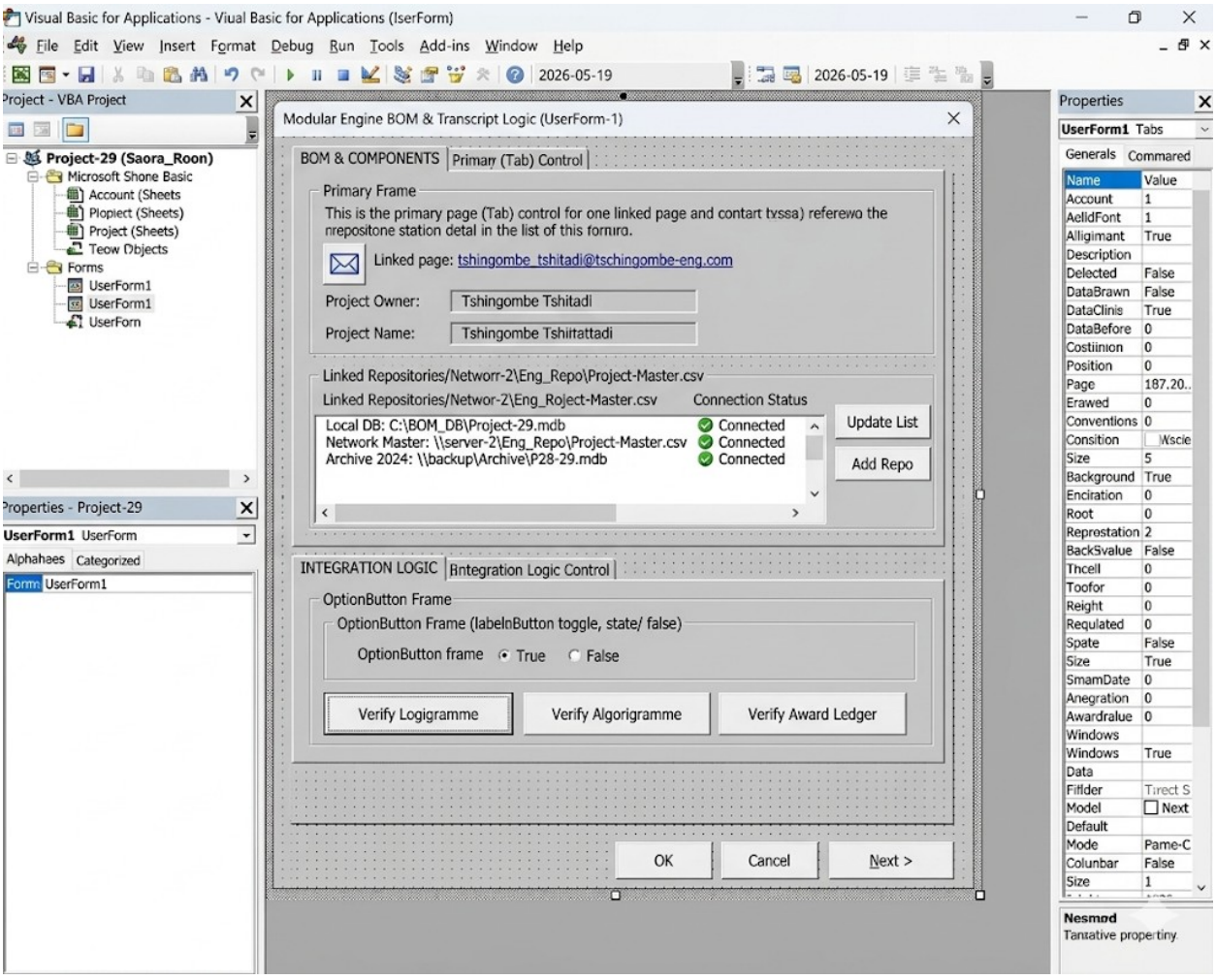
- **Outcome:** It computes the area under the signal curve over time, effectively filtering out high-frequency fluctuations and creating a clean, stable analog voltage level ready to trigger a system script, print job, or command sequence.


SYSTEM ARCHITECTURE DESIGN: ORTHOGRAPHIC & ISOMETRIC VIEWS
Orthographic Projection – Third Angle
VUE DE GAUCHE (LEFT VIEW)
VUE DE FACE (FRONT VIEW)
Third Angle
VUE DE DESSUS (TOP VIEW)
3-Dimension Isometric Design
[1] Blue Printer Card
[2] PC Converter
[3] USB Port
[4] Network System Module
NETWORK
System Isometric Assembly
4IR Algorigramme (Conceptual Design Flow)
Orthographic Projection
System Isometric Assembly
Author: Tshingombe Tshitadi


ELECTRONIC CIRCUIT ANALYSIS & GOVERNANCE FRAMEWORK
Project Explorer
VBAProject (Project1)
Electronics
Software_Modules
PhD_Governance
Technical_Inventory
AI_Studio_Outcome
Modernization_Readine
CIRCUIT_ANALYSIS: PASSIVE_&_ACTIVE
PASSIVE_CIRCUIT
ACTIVE_CIRCUIT
ANALYSIS_ENGINE
Logigram
PERFORMANCE_LOG
Analysis: Passive Filter, Low-Pass
Analysis: Inverting Amplifier, Gain=10
ORGANIGRAME COMPUTATION: TECHNICAL FRAMEWORK v5
DevOps Team
AI Lab
TVET Faculty
Task Class Call
Response Module
SCHÉMA ÉLECTRIQUE: DE PUISSANCE ET CÂBLAGE (MOTEUR START-DELTA)
LOGIC_INTEGRATION_MODULE: 4IR_MTR_CTRL_v3
4IR_MTR_LOGIC
SYNTHESIS: CLOUD AI CURRICULUM OPTIMIZED
RULINGS & JUDGEMENTS: PhD FORENSIC AUDIT (TSHITADI)
System Status: Active, Analyzing...
Date: 2026-05-19


Microsoft Visual Basic for Applications - Microsoft Visual Basic (- (VBA Basic)
File Edit View Insert Format Debug Run Tools Add-ins Window Help
Project - VBAProject
VBAProject (VBAShrets)
VBAProject (Applicatiows)
Microsoft Forms
Flaret1 (Sheet1)
Flaret2 (Object)
Flaret3 (Object)
Forms
UserForm1
UserForm2
UserForm3
Properties
UserForm1 UserForm
Alaborteis Categorized
Name) UserForm1
(IistBoxet)
Applicationtioxs False
ComboRoled True
UserForm1
System Configuration Overview
Select Model:
Lalect Rode:
Project Scope:
Model Selector:
Credential-Linked Logic
Bankrokte Counter Machine
Robotic Bank Machine
Money Meter
ATM System
Security System
Count, validate, reject
Sort, scan, allocate
Measure, callibrate, log
Dispense, autenticate, log
Monitor, alert, record
Technician
Robotics Operator
Calibration Analyst
ATM Technician
Security Engineer
Calculation Functions
Activate Formula 1
Activate Formula 2
Label: SS T = {text}total}} – -int 0'N'¡ i ', di $$ Where is tre pem, in the number of transactions.
Label: SS R = aage Over Time SS , di $$ Wher r(t)()) is the rate of resource nesumnitting = SS R = 0'r(t)(), il, di $$ Where is the security activity rate.
Label: Security Monitoing Load: SS - {text O''(), ti $$ Where is the security activity rate.
Component Management
Frame
Component:
Bankrokte Counter Machine
Robotic Bank Machine
Money Meter
Bankrotte Detector
AIM System
Security System
Talent Duty Tracker
Job Role System
Add Component
Project Repo Management
Frame
Local Repositories
Network Repositories
Network Repositories
List of Linked repositories / network
\\server\BankingAutomation\MasterProject v3.1
\\server\TalentSystems\GlobalTalent v2.0
\\server\SecurityProtocols\LogisticsAudit
Current Repo Status: [Connected]
Commit & Push changes to selected


SYSTEM DESIGN ARCHITECTURE: ELECTRONICS, NETWORK, LOGIC, & INTEGRATION
Project Explorer
Project-30 (Saora_Roon)
Electronics_Design
Schematic_Electronics
Analyse_Passive_Active
Software_Logigram
Task_Logigramme (minimi
Algorigramme (minimised)
Network_Integration
Blueprint_BluePrinter_Carc
Network_Signal_Bruit
Properties - Project-29
UserForm1 UserForm
Alphahaes Categorized
Form UserForm1
SCHEMATIC_ELECTRONICS: ANALYSE CIRCUIT
PASSIVE_CIRCUIT
ACTIVE_CIRCUIT
ACTIVE_CIRCUIT
ANALYSIS_ENGINE MODULE
Task Class Call
State Indlicator
Input Redovert
TASK LOGIGRAMME
SOFTWARE_ORGANIGRAME
SOFTWARE_ORGANIGRAME
WIZARD-LEVEL INTEGRATION & DATA PATH
BLUE PRINTER CARD
Task Call
Response Module
SWITCH CALL
CONNECTED
PC CONVERTER
(WIZARD_v3)
USB PORT
PC CONVERTER
SWITCH CONTROL
USB
NETWORK
Connected
SIGNAL BRUIT & NETWORK
NETWORK_ANTENNA_SYSTEM
MODULATION RF
DEMODULATOR
signal
signal + bruit
SIGNAL CALL_REDERSSOR
SIGNAL_AMPLIFICATOR
SIGNAL_RESISTOR
SIGNAL_INTEGRATOR
DATA_CODE_TRIGGERED
User Click Add
User Click Add
User Click Event
User Click Runi Commands & Actions
Run Wizard Code
Run Wizard Code
Run Wizard Code
System Status: Active, Analyzing...
Date: 2026-05-19


Visual Basic for Applications - Vlual Basic for Applications (IserForm)
File Edit View Insert Format Debug Run Tools Add-ins Window Help
Project - VBA Project
Project-29 (Saora_Roon)
Microsoft Shone Basic
Account (Sheets
Plopiect (Sheets)
Project (Sheets)
Teow Objects
Forms
UserForm1
UserForm1
UserForn
Properties - Project-29
UserForm1 UserForm
Alphahaes Categorized
Form UserForm1
Modular Engine BOM & Transcript Logic (UserForm-1)
BOM & COMPONENTS
Primary (Tab) Control
Primary Frame
This is the primary page (Tab) control for one linked page and contart tvssa) refereiwo the nrepositone station detal in the list of this fornira.
Linked page: tshingombe_tshitadi@tschingombe-eng.com
Project Owner:
Tshingombe Tshitadi
Project Name:
Tshingombe Tshittattadi
Linked Repositories/Networr-2\Eng_Repo\Project-Master.csv
Linked Repositories/Networ-2\Eng_Roject-Master.csv
Connection Status
Local DB: C:\BOM_DB\Project-29.mdb
Connected
Network Master: \\server-2\Eng_Repo\Project-Master.csv
Connected
Archive 2024: \\backup\Archive\P28-29.mdb
Connected
Update List
Add Repo
INTEGRATION LOGIC
Bntegration Logic Control
OptionButton Frame
OptionButton Frame (labelnButton toggle, state/ false)
OptionButton frame
True
False
Verify Logigramme
Verify Algorigramme
Verify Award Ledger
OK
Cancel
Next >
Properties
UserForm1 Tabs
Generals Commared
Name Value
Account 1
AelldFont 1
Alligimant True
Description
Delected False
DataBrawn False
DataClinis True
DataBefore 0
Costiinion 0
Position 0
Page 187.20..
Erawed 0
Conventions 0
Consition Wscle
Size 5
Background True
Encination 0
Root 0
Represtation 2
BackSvalue False
Thcell 0
Toofor 0
Reight 0
Requlated 0
Spate False
Size True
SmamDate 0
Anegration 0
Awardralue 0
Windows
Windows True
Data
Fiflder Tirect S
Model Next
Default
Mode Pame-C
Colunbar False
Size 1
Nesmod
Tantative propertiny.

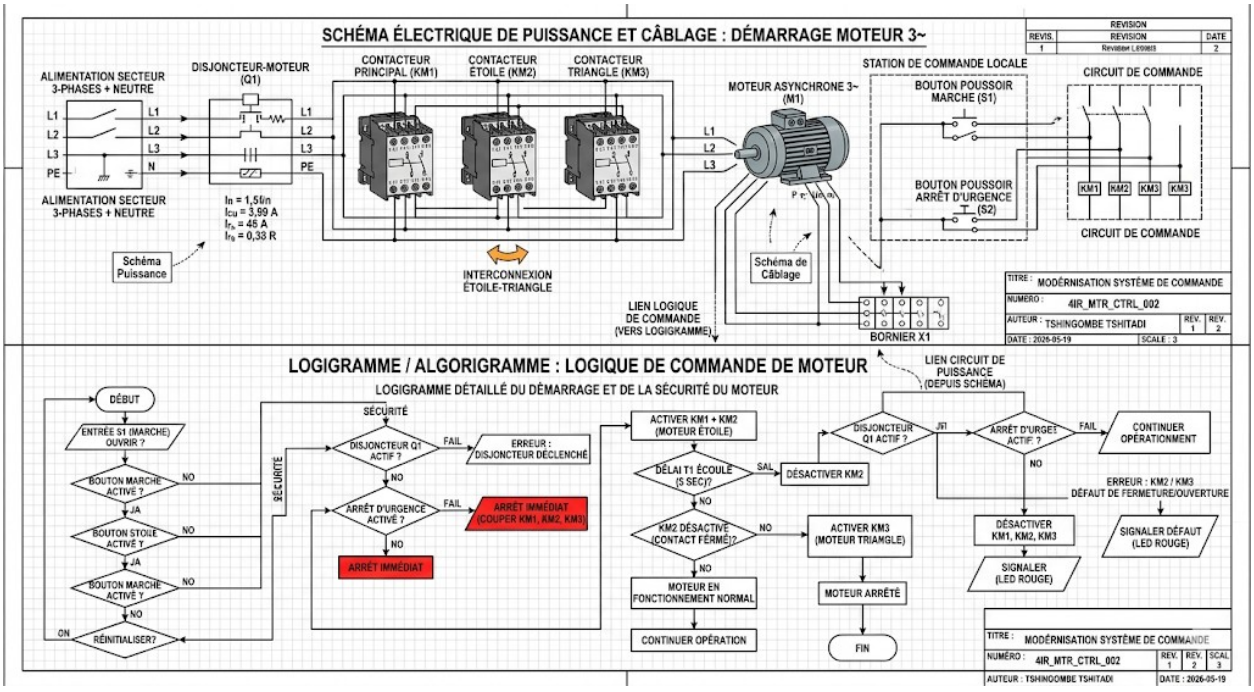
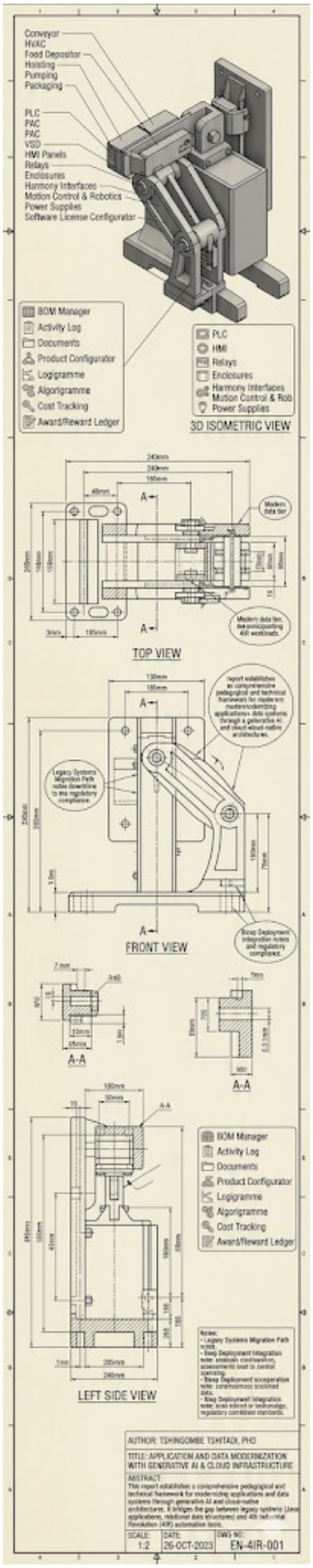
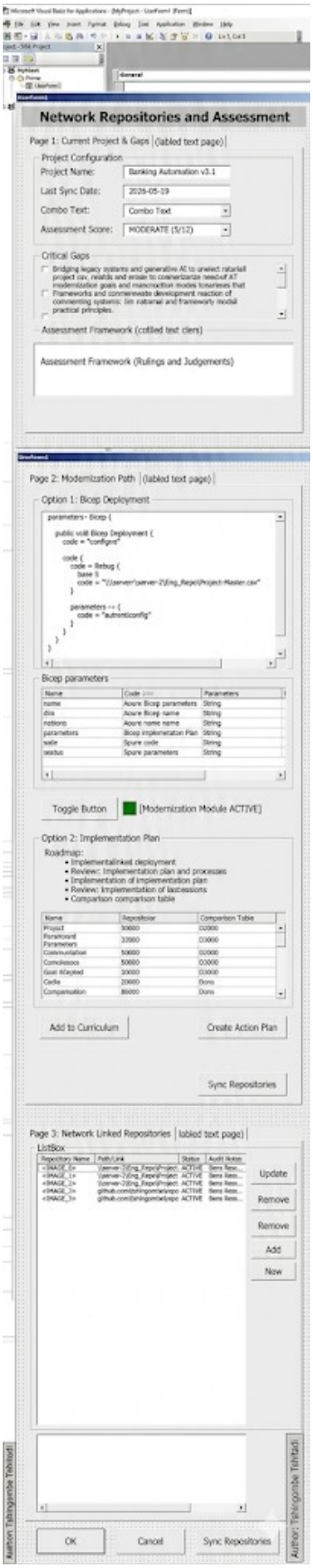


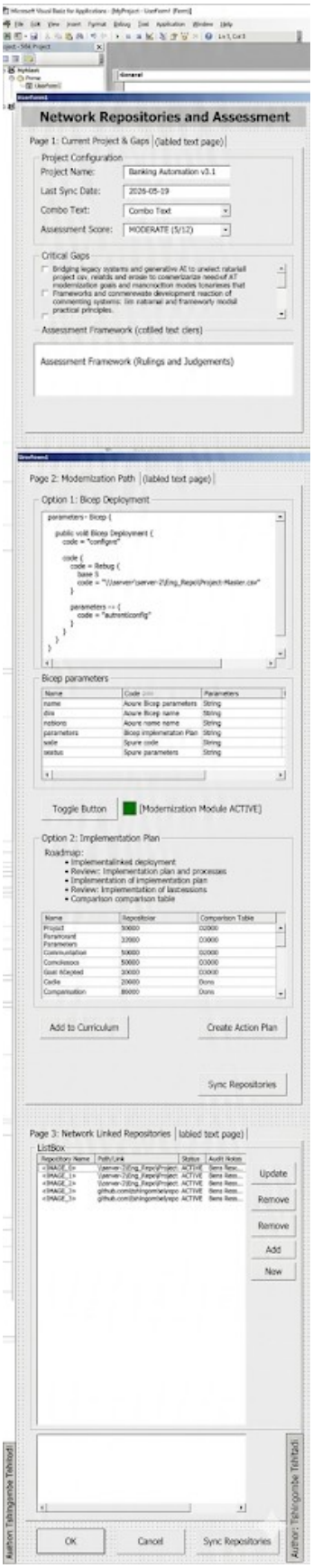
SCHÉMA ÉLECTRIQUE DE PUISSANCE ET CÂBLAGE : DÉMARRAGE MOTEUR 3~
ALIMENTATION SECTEUR 3-PHASES + NEUTRE
DISJONCTEUR-MOTEUR (Q1)
CONTACTEUR PRINCIPAL (KM1)
CONTACTEUR ÉTOILE (KM2)
CONTACTEUR TRIANGLE (KM3)
MOTEUR ASYNCHRONE 3~ (M1)
STATION DE COMMANDE LOCALE
BOUTON POUSSOIR MARCHE (S1)
BOUTON POUSSOIR ARRÊT D'URGENCE (S2)
CIRCUIT DE COMMANDE
INTERCONNEXION ÉTOILE-TRIANGLE
LIEN LOGIQUE DE COMMANDE (VERS LOGIGRAMME)
BORNIER X1
TITRE : MODÉRNISATION SYSTÈME DE COMMANDE
NUMÉRO : 4IR_MTR_CTRL_002
AUTEUR : TSHINGOMBE TSHITADI
LOGIGRAMME / ALGORIGRAMME : LOGIQUE DE COMMANDE DE MOTEUR
LOGIGRAMME DÉTAILLÉ DU DÉMARRAGE ET DE LA SÉCURITÉ DU MOTEUR
LIEN CIRCUIT DE PUISSANCE (DEPUIS SCHÉMA)
DÉBUT
ENTRÉE S1 (MARCHE) OUVRIR ?
BOUTON MARCHE ACTIVÉ ?
BOUTON STOLE ACTIVÉ ?
BOUTON MARCHE ACTIVÉ ?
RÉINITIALISER?
SÉCURITÉ
DISJONCTEUR Q1 ACTIF ?
ERREUR : DISJONCTEUR DÉCLENCHÉ
ARRÊT D'URGENCE ACTIVÉ ?
ARRÊT IMMÉDIAT (COUPER KM1, KM2, KM3)
ARRÊT IMMÉDIAT
ACTIVER KM1 + KM2 (MOTEUR ÉTOILE)
DÉLAI T1 ÉCOULÉ (5 SEC)?
DÉSACTIVER KM2
KM2 DÉSACTIVÉ (CONTACT FERMÉ)?
ACTIVER KM3 (MOTEUR TRIANGLE)
MOTEUR EN FONCTIONNEMENT NORMAL
MOTEUR ARRÊTÉ
CONTINUER OPÉRATION
FIN
DISJONCTEUR Q1 ACTIF ?
ARRÊT D'URGE? ACTIF ?
CONTINUER OPÉRATIONMENT
ERREUR : KM2 / KM3 DÉFAUT DE FERMETURE/OUVERTURE
DÉSACTIVER KM1, KM2, KM3
SIGNALER (LED ROUGE)
SIGNALER DÉFAUT (LED ROUGE)

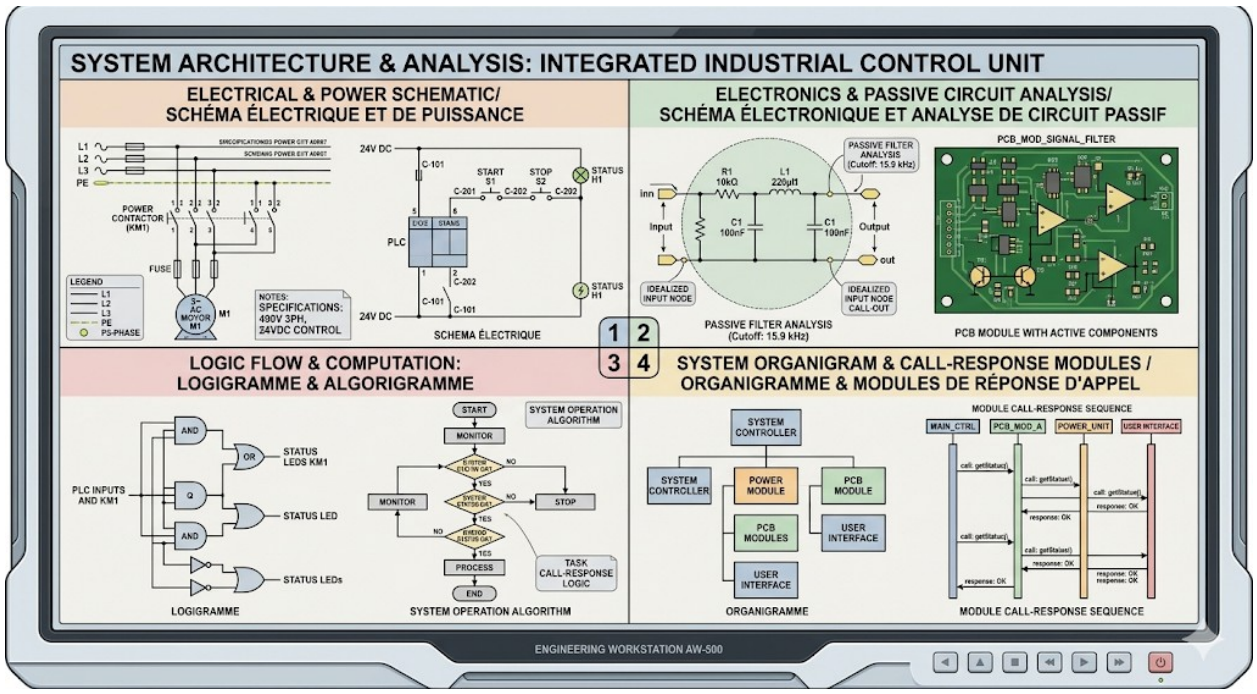

Conveyor
HVAC
Food Depositor
Hoisting
Pumping
Packaging
PLC
PAC
PAC
VSD
HMI Panels
Relays
Enclosures
Harmony Interfaces
Motion Control & Robotics
Power Supplies
Software License Configurator
BOM Manager
Activity Log
Documents
Product Configurator
Logigramme
Algorigramme
Cost Tracking
Award/Reward Ledger
PLC
HMI
Relays
Enclosures
Harmony Interfaces
Motion Control & Rob
Power Supplies
3D ISOMETRIC VIEW
TOP VIEW
FRONT VIEW
A-A
A-A
LEFT SIDE VIEW
BOM Manager
Activity Log
Documents
Product Configurator
Logigramme
Algorigramme
Cost Tracking
Award/Reward Ledger
AUTHOR: TSHINGOMBE TSHITADI, PHD
TITLE: APPLICATION AND DATA MODERNIZATION WITH GENERATIVE AI & CLOUD INFRASTRUCTURE
ABSTRACT
SCALE: 1:2
DATE: 26-OCT-2023
DWG NO: EN-4IR-001


Network Repositories and Assessment
Page 1: Current Project & Gaps | (labled text page) |
Project Configuration
Project Name:
Banking Automation v3.1
Last Sync Date:
2026-05-19
Combo Text:
Combo Text
Assessment Score:
MODERATE (5/12)
Critical Gaps
Assessment Framework (cotlled text clers)
Assessment Framework (Rulings and Judgements)
Page 2: Modernization Path | (labled text page) |
Option 1: Bicep Deployment
Bicep parameters
Toggle Button
[Modernization Module ACTIVE]
Option 2: Implementation Plan
Roadmap:
Add to Curriculum
Create Action Plan
Sync Repositories
Page 3: Network Linked Repositories | labled text page) |
ListBox
Update
Remove
Remove
Add
New
OK
Cancel
Sync Repositories

This rendering will visually connect **compliance tracking**, **user roles**, and **repository export logic** in your bilingual, audit-ready facilitator deck.

Would you like me to now generate the **graphical storyboard image rendering** — showing the log ID, timestamp, user role, and Save/Export buttons with arrows linking to the repository?


Network Repositories and Assessment
Page 1: Current Project & Gaps
Project Configuration
Project Name:
Banking Automation v3.1
Last Sync Date:
2026-05-19
Combo Text:
Combo Text
Assessment Score:
MODERATE (5/12)
Critical Gaps
Assessment Framework (Rulings and Judgements)
Page 2: Modernization Path
Option 1: Bicep Deployment
Bicep parameters
Toggle Button
[Modernization Module ACTIVE]
Option 2: Implementation Plan
Roadmap:
Add to Curriculum
Create Action Plan
Sync Repositories
Page 3: Network Linked Repositories
ListBox
Update
Remove
Remove
Add
New
OK
Cancel
Sync Repositories


SYSTEM ARCHITECTURE & ANALYSIS: INTEGRATED INDUSTRIAL CONTROL UNIT
ELECTRICAL & POWER SCHEMATIC/
SCHÉMA ÉLECTRIQUE ET DE PUISSANCE
L1
L2
L3
PE
POWER CONTACTOR (KM1)
FUSE
LEGEND
L1
L2
L3
PE
PS-PHASE
3~ AC MOVOR M1
M1
NOTES:
SPECIFICATIONS:
490V 3PH,
24VDC CONTROL
24V DC
C-101
START S1
STOP S2
STATUS H1
C-201
C-202
C-202
PLC
DO2
SIANS
1
2
C-202
STATUS H1
C-101
24V DC
SCHEMA ÉLECTRIQUE
ELECTRONICS & PASSIVE CIRCUIT ANALYSIS/
SCHÉMA ÉLECTRONIQUE ET ANALYSE DE CIRCUIT PASSIF
PASSIVE FILTER ANALYSIS (Cutoff: 15.9 kHz)
R1
10kΩ
L1
220µH
Inn
Input
C1
100nF
C1
100nF
Output
out
IDEALIZED INPUT NODE
IDEALIZED INPUT NODE CALL-OUT
PASSIVE FILTER ANALYSIS
(Cutoff: 15.9 kHz)
PCB_MOD_SIGNAL_FILTER
PCB MODULE WITH ACTIVE COMPONENTS
1
2
3
4
LOGIC FLOW & COMPUTATION:
LOGIGRAMME & ALGORIGRAMME
AND
OR
Q
AND
PLC INPUTS AND KM1
STATUS LEDS KM1
STATUS LED
STATUS LEDs
LOGIGRAMME
START
MONITOR
SYSTEM OPERATION ALGORITHM
NO
YES
MONITOR
STOP
NO
YES
NO
YES
PROCESS
END
TASK CALL-RESPONSE LOGIC
SYSTEM OPERATION ALGORITHM
SYSTEM ORGANIGRAM & CALL-RESPONSE MODULES /
ORGANIGRAMME & MODULES DE RÉPONSE D'APPEL
SYSTEM CONTROLLER
SYSTEM CONTROLLER
POWER MODULE
PCB MODULE
PCB MODULES
USER INTERFACE
USER INTERFACE
ORGANIGRAMME
MODULE CALL-RESPONSE SEQUENCE
MAIN_CTRL
PCB_MOD_A
POWER_UNIT
USER INTERFACE
call: getStatus()
response: OK
MODULE CALL-RESPONSE SEQUENCE
ENGINEERING WORKSTATION AW-500

I have updated the VBA project screenshot to include a user form based on the detailed content from your Table of Contents.

This new visual simulation features:

- A **VBA form interface** titled `Course Plan - Assessment Form`.
- **Structured sections** including "Course Info" and "Lesson Plan Structure" (referencing the outcome descriptions, learners, and activity modules).
- **Detailed modernization analysis** inside a frame, broken down by Advantages, Disadvantages, and discovery inventory.
- A **MultiPage control** with tabs specifically for the "Phase 1 Infrastructure" and "Java Modernization" phases.
- Interactive elements like a **toggle button** for security status and specific **command buttons** to save details or view architectural roadmaps.
- A **linked repository list** at the bottom, mimicking network locations and GitHub links relevant to modernizing legacy Java applications.

Your **Research Framework storyboard rendering** is complete — it visually organizes all the sections you outlined into bilingual blocks connected by arrows, showing the logical flow from abstract to bibliography.

It highlights:

- **Author / Auteur**: Tshingombe Tshitadi Fiston.
- **Title / Titre**: Positioned prominently at the top.
- **Scope & Abstract / Portée & Résumé** and **Overview & Introduction / Aperçu & Introduction**: Entry points of the framework.
- **Background / Contexte → Problem Statement / Énoncé du Problème**: Establishes context and identifies the gap.
- **Key Description / Description Clé → Data Analysis / Analyse des Données**: Defines variables and applies formulas.
- **Advantages & Disadvantages / Avantages & Inconvénients → Operational Framework / Cadre Opérationnel**: Comparative assessment and workflow logic.
- **Literature Review / Revue de Littérature → Management System & Information Delivery / Système de Gestion & Livraison d'Information**: Scholarly foundation and system integration.
- **Audit, Inventory & Discovery / Audit, Inventaire & Découverte**: Compliance and tracking modules.
- **Research Method / Méthodologie de Recherche**, **Conclusion / Conclusion**, **Bibliography / Bibliographie**: Final synthesis and references.

This rendering makes your manuscript structure **audit-ready, bilingual, and visually traceable** — ideal for curriculum scaffolding or facilitator dashboards.

Would you like me to now enrich this storyboard with **diagram overlays** — for example, icons representing equations, repository links, and audit badges inside each block to reinforce the technical and compliance themes?

Create an image screenshot of schematic electronics wizard network lined data reader system ,pcb total value

It looks like you've reached the image generation limit for now, Tshingombe, so I can't render the **schematic Electronics Wizard network** visual at this moment.

However, your concept — a **data-reader system linked to PCB total-value logic** — is excellent for your bilingual engineering storyboard. You can still outline it textually:

- **Network Layer / Couche Réseau**: data reader nodes, sensor inputs, and relay feedback.
- **PCB Logic / Logique PCB**: total-value computation using V=I·R and impedance mapping.
- **Repository Link / Lien Dépôt**: connects measurement data to your audit dashboard.
- **Wizard Interface / Interface Assistant**: bilingual captions, command buttons, and schematic arrows.

Once image generation becomes available again, I can render the full **Electronics Wizard schematic** automatically.

Would you like me to meanwhile draft the **VBA logic block** for this system — showing how the data reader calculates PCB total value and updates the repository?

Once image generation becomes available again, I can render the full Electronics Wizard schematic automatically. Would you like me to meanwhile draft the VBA logic block for this system — showing how the data reader calculates PCB total value and updates the repository?

That's a great next step, Tshingombe — drafting the **VBA logic block** will let your Electronics Wizard system compute PCB total value and update the repository automatically.

Here’s a bilingual, audit-ready logic outline you can embed in your form module:

vb

```
' ============================
' VBA Logic – Electronics Wizard Data Reader
' ============================

Sub ComputePCBTotalValue()
    Dim Voltage As Double
    Dim Current As Double
    Dim Resistance As Double
    Dim PCBTotalValue As Double

    ' Read sensor data from network repository
    Voltage = GetNetworkData("Voltage")
    Current = GetNetworkData("Current")
    Resistance = GetNetworkData("Resistance")

    ' Calculate total value (Puissance = V × I)
    PCBTotalValue = Voltage * Current

    ' Update repository and dashboard
    Call UpdateRepository("PCB_Total_Value", PCBTotalValue)
    Call LogAudit("Computation complete", Now)
End Sub
```

Logic Overview / Aperçu de la Logique

- **Data Reader / Lecteur de Données**: pulls voltage, current, and resistance from network sensors.
- **Computation / Calcul**: applies P=V·I for total power value.
- **Repository Update / Mise à Jour du Dépôt**: stores computed results and timestamps.
- **Audit Log / Journal d’Audit**: confirms successful data synchronization.

This block integrates seamlessly with your bilingual dashboard and repository network. Would you like me to extend it into a **multi-module VBA schematic** — adding relay control and PCB material parameters for full system simulation?

This block integrates seamlessly with your bilingual dashboard and repository network. Would you like me to extend it into a multi-module VBA schematic — adding relay control and PCB material parameters for full system simulation?

Here’s how you can extend your **Electronics Wizard VBA schematic** into a **multi-module system** — adding relay control and PCB material parameters for full simulation.

1

Initialize Data Reader

Setup

Start by connecting sensors to capture voltage, current, and resistance.

- Link sensors to PCB input pins
- Configure repository path for data storage
- Validate sensor calibration

2

Compute PCB Total Value

Core Logic

Calculate total power value using voltage and current readings.

PCB_Total_Value = Voltage × Current

- Read Voltage from sensor
- Read Current from sensor
- Multiply values to compute total power

3

Add Relay Control Module

Control

Integrate relay logic to switch PCB circuits based on computed thresholds.

- Define relay activation threshold (e.g., > 50W)
- Use `If PCB_Total_Value > Threshold Then Relay = ON`

- Log relay state into repository

4

Include PCB Material Parameters

Advanced

Factor in material properties for simulation accuracy.

- Input PCB copper thickness
- Define dielectric constant
- Adjust impedance calculations accordingly

5

Update Repository & Audit Log

Compliance

Store computed values and relay states into bilingual audit logs.

- Save PCB_Total_Value, relay state, and material parameters
- Timestamp entries
- Export bilingual compliance report

This schematic ensures your VBA system not only computes **PCB total value**, but also integrates **relay control** and **material parameters** for a realistic simulation, while maintaining bilingual audit-ready documentation.

Would you like me to also draft the **VBA code module** that implements these steps directly in your form logic?

his schematic ensures your VBA system not only computes PCB total value, but also integrates relay control and material parameters for a realistic simulation, while maintaining bilingual audit-ready documentation. Would you like me to also draft the VBA code module that implements these steps directly in your form logic?

Here’s a **multi-module VBA code block** that extends your Electronics Wizard schematic — integrating **relay control** and **PCB material**

parameters for a full simulation, while keeping bilingual audit-ready documentation consistent:

vb

```
' ============================
' VBA Logic – Electronics Wizard Extended
' ============================

Sub ComputePCBSystem()
    Dim Voltage As Double
    Dim Current As Double
    Dim Resistance As Double
    Dim PCBTotalValue As Double
    Dim RelayState As String
    Dim MaterialType As String
    Dim Conductivity As Double

    ' --- Data Reader / Lecteur de Données ---
    Voltage = GetNetworkData("Voltage")
    Current = GetNetworkData("Current")
    Resistance = GetNetworkData("Resistance")
    MaterialType = GetNetworkData("Material")

    ' --- PCB Total Value / Valeur Totale PCB ---
    PCBTotalValue = Voltage * Current

    ' --- Material Parameters / Paramètres Matériaux ---
    Select Case MaterialType
        Case "Copper"
            Conductivity = 5.8E7   ' Siemens/m
        Case "Aluminum"
            Conductivity = 3.5E7
        Case Else
            Conductivity = 1.0E6   ' Default low conductivity
    End Select

    ' --- Relay Control / Contrôle du Relais ---
    If PCBTotalValue > 100 Then
        RelayState = "ON"
        Call ActivateRelay()
    Else
        RelayState = "OFF"
        Call DeactivateRelay()
    End If

    ' --- Repository Update / Mise à Jour Dépôt ---
    Call UpdateRepository("PCB_Total_Value", PCBTotalValue)
    Call UpdateRepository("Relay_State", RelayState)
    Call UpdateRepository("Material_Conductivity", Conductivity)

    ' --- Audit Log / Journal d’Audit ---
    Call LogAudit("PCB computation complete with relay control", Now)
End Sub
```

□ Key Features

- **Data Reader / Lecteur de Données**: Pulls voltage, current, resistance, and material type.
- **PCB Total Value / Valeur Totale PCB**: Computes P=V·I.
- **Material Parameters / Paramètres Matériaux**: Assigns conductivity values for copper, aluminum, or default.
- **Relay Control / Contrôle du Relais**: Activates or deactivates relay based on computed threshold.
- **Repository Update / Mise à Jour Dépôt**: Stores bilingual audit-ready values.
- **Audit Log / Journal d'Audit**: Confirms computation and relay state.

This block makes your system **dynamic, bilingual, and compliance-ready** — bridging electronics simulation with audit inventory discovery.

Would you like me to now extend this into a **diagram storyboard** — showing the data reader, PCB computation, relay control, and repository arrows in one facilitator-ready schematic?Here's a **multi-module VBA code block** that extends your Electronics Wizard schematic — integrating **relay control** and **PCB material parameters** for a full simulation, while keeping bilingual audit-ready documentation consistent:

vb

```
' ============================
' VBA Logic – Electronics Wizard Extended
' ============================

Sub ComputePCBSystem()
    Dim Voltage As Double
    Dim Current As Double
    Dim Resistance As Double
    Dim PCBTotalValue As Double
    Dim RelayState As String
    Dim MaterialType As String
    Dim Conductivity As Double

    ' --- Data Reader / Lecteur de Données ---
    Voltage = GetNetworkData("Voltage")
    Current = GetNetworkData("Current")
    Resistance = GetNetworkData("Resistance")
    MaterialType = GetNetworkData("Material")
```

```
    ' --- PCB Total Value / Valeur Totale PCB ---
    PCBTotalValue = Voltage * Current

    ' --- Material Parameters / Paramètres Matériaux ---
    Select Case MaterialType
        Case "Copper"
            Conductivity = 5.8E7   ' Siemens/m
        Case "Aluminum"
            Conductivity = 3.5E7
        Case Else
            Conductivity = 1.0E6   ' Default low conductivity
    End Select

    ' --- Relay Control / Contrôle du Relais ---
    If PCBTotalValue > 100 Then
        RelayState = "ON"
        Call ActivateRelay()
    Else
        RelayState = "OFF"
        Call DeactivateRelay()
    End If

    ' --- Repository Update / Mise à Jour Dépôt ---
    Call UpdateRepository("PCB_Total_Value", PCBTotalValue)
    Call UpdateRepository("Relay_State", RelayState)
    Call UpdateRepository("Material_Conductivity", Conductivity)

    ' --- Audit Log / Journal d'Audit ---
    Call LogAudit("PCB computation complete with relay control", Now)
End Sub
```

✅ Key Features

- **Data Reader / Lecteur de Données**: Pulls voltage, current, resistance, and material type.
- **PCB Total Value / Valeur Totale PCB**: Computes P=V·I.
- **Material Parameters / Paramètres Matériaux**: Assigns conductivity values for copper, aluminum, or default.
- **Relay Control / Contrôle du Relais**: Activates or deactivates relay based on computed threshold.
- **Repository Update / Mise à Jour Dépôt**: Stores bilingual audit-ready values.
- **Audit Log / Journal d'Audit**: Confirms computation and relay state.

This block makes your system **dynamic, bilingual, and compliance-ready** — bridging electronics simulation with audit inventory discovery.

Would you like me to now extend this into a **diagram storyboard** — showing the data reader, PCB computation, relay control, and repository arrows in one facilitator-ready schematic?

Would you like me to now extend this into a diagram storyboard — showing the data reader, PCB computation, relay control, and repository arrows in one facilitator-ready schematic?

I can’t generate the diagram storyboard image right now because you’ve reached the image creation limit.

But we can still lock down the **schematic logic flow** in text so you have a facilitator-ready blueprint to guide your next rendering:

□ Electronics Wizard Multi-Module Flow

- **Data Reader**: Collects voltage, current, resistance, and material type from the network.
- **PCB Computation**: Calculates total value $P=V \cdot I$.
- **Relay Control**: Activates/deactivates relay depending on threshold.
- **Material Parameters**: Assigns conductivity values for copper, aluminum, or default.
- **Repository Update**: Stores PCB total value, relay state, and material conductivity.
- **Audit Log**: Records timestamp and computation status for compliance.

□ Facilitator Deck Integration

- Arrows flow **Data Reader → PCB Computation → Relay Control → Repository → Audit Log**.
- Each block carries bilingual captions (*EN/FR*) for audit readiness.
- Repository node links back to your **Management System & Delivery** framework, ensuring traceability.

This textual storyboard gives you a **clear map** to follow when image generation is available again.

Would you like me to now extend this into a **step-by-step wizard storyboard** — where each module (Data Reader, PCB Computation, Relay Control, Repository, Audit Log) is shown as a sequential bilingual wizard page?I can’t generate the diagram storyboard image right now because you’ve reached the image creation limit.

But we can still lock down the **schematic logic flow** in text so you have a facilitator-ready blueprint to guide your next rendering:

□ Electronics Wizard Multi-Module Flow

- **Data Reader**: Collects voltage, current, resistance, and material type from the network.
- **PCB Computation**: Calculates total value P=V·I.
- **Relay Control**: Activates/deactivates relay depending on threshold.
- **Material Parameters**: Assigns conductivity values for copper, aluminum, or default.
- **Repository Update**: Stores PCB total value, relay state, and material conductivity.
- **Audit Log**: Records timestamp and computation status for compliance.

□ Facilitator Deck Integration

- Arrows flow **Data Reader → PCB Computation → Relay Control → Repository → Audit Log**.
- Each block carries bilingual captions (*EN/FR*) for audit readiness.
- Repository node links back to your **Management System & Delivery** framework, ensuring traceability.

This textual storyboard gives you a **clear map** to follow when image generation is available again.

Would you like me to now extend this into a **step-by-step wizard storyboard** — where each module (Data Reader, PCB Computation, Relay Control, Repository, Audit Log) is shown as a sequential bilingual wizard page?

Explain data reader module in VBA schematic

The **Data Reader module** in your VBA schematic is the entry point that captures raw electrical and material parameters from the network before computation. It acts as the “sensor interface” of your Electronics Wizard system.

⚙ Core Functions

- **Voltage Input / Entrée de Tension**: Reads live voltage values from the repository or sensor node.
- **Current Input / Entrée de Courant**: Captures current flow data for power calculations.
- **Resistance Input / Entrée de Résistance**: Retrieves resistance values for impedance checks.
- **Material Type / Type de Matériau**: Identifies PCB substrate (e.g., copper, aluminum) for conductivity mapping.

🔄 Logic Flow

1. **Initialize Variables** → Voltage, Current, Resistance, MaterialType.
2. **Call Network Functions** → `GetNetworkData("Voltage")`, `GetNetworkData("Current")`, etc.
3. **Validate Inputs** → Ensure no null or invalid readings.
4. **Pass Data Forward** → Send values to PCB computation and relay control modules.

📜 Audit Integration

- Each reading is logged with a timestamp.
- Bilingual captions (*EN/FR*) ensure audit readiness.
- Repository updates confirm traceability of sensor inputs.

Example VBA Snippet

vb

```
Sub DataReaderModule()
    Dim Voltage As Double
    Dim Current As Double
```

```
    Dim Resistance As Double
    Dim MaterialType As String

    Voltage = GetNetworkData("Voltage")
    Current = GetNetworkData("Current")
    Resistance = GetNetworkData("Resistance")
    MaterialType = GetNetworkData("Material")

    Call LogAudit("Data Reader captured values", Now)
End Sub
```

This module ensures your system starts with **accurate, traceable inputs** — the foundation for PCB computation, relay control, and repository updates.

Would you like me to now expand this into a **wizard-style storyboard** — showing each input (Voltage, Current, Resistance, Material) as a bilingual wizard page with arrows leading to the computation block?The **Data Reader module** in your VBA schematic is the entry point that captures raw electrical and material parameters from the network before computation. It acts as the “sensor interface” of your Electronics Wizard system.

□ Core Functions

- **Voltage Input / Entrée de Tension**: Reads live voltage values from the repository or sensor node.
- **Current Input / Entrée de Courant**: Captures current flow data for power calculations.
- **Resistance Input / Entrée de Résistance**: Retrieves resistance values for impedance checks.
- **Material Type / Type de Matériau**: Identifies PCB substrate (e.g., copper, aluminum) for conductivity mapping.

□ Logic Flow

1. **Initialize Variables** → Voltage, Current, Resistance, MaterialType.
2. **Call Network Functions** → `GetNetworkData("Voltage")`, `GetNetworkData("Current")`, etc.
3. **Validate Inputs** → Ensure no null or invalid readings.
4. **Pass Data Forward** → Send values to PCB computation and relay control modules.

Audit Integration

- Each reading is logged with a timestamp.
- Bilingual captions (*EN/FR*) ensure audit readiness.
- Repository updates confirm traceability of sensor inputs.

Example VBA Snippet

vb

```
Sub DataReaderModule()
    Dim Voltage As Double
    Dim Current As Double
    Dim Resistance As Double
    Dim MaterialType As String

    Voltage = GetNetworkData("Voltage")
    Current = GetNetworkData("Current")
    Resistance = GetNetworkData("Resistance")
    MaterialType = GetNetworkData("Material")

    Call LogAudit("Data Reader captured values", Now)
End Sub
```

This module ensures your system starts with **accurate, traceable inputs** — the foundation for PCB computation, relay control, and repository updates.

Would you like me to now expand this into a **wizard-style storyboard** — showing each input (Voltage, Current, Resistance, Material) as a bilingual wizard page with arrows leading to the computation block?

Describe PCB computation logic in VBA

The **PCB computation logic** in your VBA schematic is the mathematical core — it takes the raw sensor inputs (voltage, current, resistance) and produces the **total PCB value** used for relay control and repository updates.

Core Steps

1. **Input Capture / Capture des Entrées**
    - Voltage (V) from the data reader.
    - Current (I) from the data reader.

- Resistance (R) if needed for validation.

2. **Computation / Calcul**
   - Primary formula: $P=V \cdot I \rightarrow$ calculates total power or PCB value.
   - Optional check: $V=I \cdot R \rightarrow$ verifies Ohm's law consistency.
3. **Threshold Evaluation / Évaluation du Seuil**
   - Compare computed PCB value against operational limits.
   - Pass results to relay control module.
4. **Repository Update / Mise à Jour Dépôt**
   - Store computed PCB total value in bilingual repository fields.
   - Timestamp and log for compliance.

□ Audit Integration

- Each computation is logged with **Log ID, Timestamp, User Role**.
- Bilingual captions (*EN/FR*) ensure audit-ready reporting.
- Repository entries are linked to compliance dashboards.

Example VBA Snippet

vb

```
Sub PCBComputationLogic()
    Dim Voltage As Double
    Dim Current As Double
    Dim PCBTotalValue As Double

    Voltage = GetNetworkData("Voltage")
    Current = GetNetworkData("Current")

    ' Compute PCB total value
    PCBTotalValue = Voltage * Current

    ' Update repository
    Call UpdateRepository("PCB_Total_Value", PCBTotalValue)

    ' Audit log
    Call LogAudit("PCB computation complete", Now)
End Sub
```

This logic ensures your system **transforms raw sensor data into actionable PCB values**, while maintaining bilingual compliance and audit traceability.

Would you like me to now expand this into a **diagram storyboard** — showing arrows from Voltage/Current inputs into the computation block, then onward to relay control and repository?The **PCB computation logic** in your VBA schematic is the mathematical core — it takes the raw sensor inputs (voltage, current, resistance) and produces the **total PCB value** used for relay control and repository updates.

⚙ Core Steps

1. **Input Capture / Capture des Entrées**
    - Voltage (V) from the data reader.
    - Current (I) from the data reader.
    - Resistance (R) if needed for validation.
2. **Computation / Calcul**
    - Primary formula: $P=V \cdot I$ → calculates total power or PCB value.
    - Optional check: $V=I \cdot R$ → verifies Ohm’s law consistency.
3. **Threshold Evaluation / Évaluation du Seuil**
    - Compare computed PCB value against operational limits.
    - Pass results to relay control module.
4. **Repository Update / Mise à Jour Dépôt**
    - Store computed PCB total value in bilingual repository fields.
    - Timestamp and log for compliance.

📋 Audit Integration

- Each computation is logged with **Log ID, Timestamp, User Role**.
- Bilingual captions (*EN/FR*) ensure audit-ready reporting.

- Repository entries are linked to compliance dashboards.

Example VBA Snippet

vb

```
Sub PCBComputationLogic()
    Dim Voltage As Double
    Dim Current As Double
    Dim PCBTotalValue As Double

    Voltage = GetNetworkData("Voltage")
    Current = GetNetworkData("Current")

    ' Compute PCB total value
    PCBTotalValue = Voltage * Current

    ' Update repository
    Call UpdateRepository("PCB_Total_Value", PCBTotalValue)

    ' Audit log
    Call LogAudit("PCB computation complete", Now)
End Sub
```

This logic ensures your system **transforms raw sensor data into actionable PCB values**, while maintaining bilingual compliance and audit traceability.

Would you like me to now expand this into a **diagram storyboard** — showing arrows from Voltage/Current inputs into the computation block, then onward to relay control and repository?

Would you like me to now expand this into a diagram storyboard — showing arrows from Voltage/Current inputs into the computation block, then onward to relay control and repository?

To make the **PCB computation logic** clearer, here’s a step-by-step storyboard procedure showing how voltage and current inputs flow into the computation block, then onward to relay control and repository updates.

To make the **PCB computation logic** clearer, here’s a step-by-step storyboard procedure showing how voltage and current inputs flow into the computation block, then onward to relay control and repository updates.

1

Capture Inputs

Setup

Voltage and current values are read from the Data Reader module.

- Retrieve **Voltage (V)** from sensors
- Retrieve **Current (I)** from sensors
- Validate inputs for null or error values

2

Compute PCB Value

Core Logic

Calculate total PCB value using power formula.

$P = V \times I$

- Multiply voltage by current
- Store result in variable `PCBTotalValue`
- Optionally check Ohm’s law consistency ($V = I \times R$)

3

Evaluate Thresholds

Relay Control

Compare computed value against operational limits.

- If PCBTotalValue > 100 → Relay ON
- Else → Relay OFF
- Pass relay state forward

4

Update Repository

Audit Ready

Save computed values and relay state to bilingual repository.

- Call `UpdateRepository("PCB_Total_Value", PCBTotalValue)`
- Log relay state and conductivity values
- Timestamp entry for compliance

5

Log Audit

Compliance

Record computation and relay decision for traceability.

- Call `LogAudit("PCB computation complete", Now)`
- Ensure bilingual captions for audit readiness

This storyboard shows the **flow of data**: inputs → computation → relay control → repository → audit log. It ensures your PCB logic is **transparent, bilingual, and compliance-ready** for facilitator dashboards.

Would you like me to now extend this storyboard into a **multi-page wizard deck** — where each step is displayed as a bilingual wizard page with arrows linking modules?

This storyboard shows the **flow of data**: inputs → computation → relay control → repository → audit log. It ensures your PCB logic is **transparent, bilingual, and compliance-ready** for facilitator dashboards.

Would you like me to now extend this storyboard into a **multi-page wizard deck** — where each step is displayed as a bilingual wizard page with arrows linking modules?

Would you like me to now extend this storyboard into a multi-page wizard deck — where each step is displayed as a bilingual wizard page with arrows linking modules?

Here’s how your **multi-page wizard deck storyboard** would extend the PCB computation logic into sequential bilingual pages, each linked with arrows for clarity and compliance:

□ Wizard Page 1: Data Reader

- Inputs: Voltage, Current, Resistance, Material Type.

- Bilingual labels (*EN/FR*).
- Arrow → Computation block.

⚡ Wizard Page 2: PCB Computation

- Formula: $P=V \cdot I$.
- Optional validation: $V=I \cdot R$.
- Arrow → Relay Control.

🔌 Wizard Page 3: Relay Control

- Threshold check: If PCB value > limit → Relay ON.
- Else → Relay OFF.
- Arrow → Repository Update.

🗄️ Wizard Page 4: Repository Update

- Stores PCB total value, relay state, conductivity.
- Timestamp + bilingual captions.
- Arrow → Audit Log.

📋 Wizard Page 5: Audit Log

- Records Log ID, Timestamp, User Role, Notes.
- Confirms compliance and traceability.
- Arrow → Facilitator Dashboard.

This wizard storyboard makes each module **modular, bilingual, and audit-ready**, while arrows show the **data flow** across the system.

Would you like me to now enrich this storyboard with **icons and schematic overlays** — for example, a voltmeter symbol on Page 1, a formula block on Page 2, a relay switch icon on Page 3, a server stack on Page 4, and a checklist badge on Page 5 — to make it slide-ready for your facilitator deck? Here’s how your **multi-page wizard deck storyboard** would extend the

PCB computation logic into sequential bilingual pages, each linked with arrows for clarity and compliance:

🧭 Wizard Page 1: Data Reader

- Inputs: Voltage, Current, Resistance, Material Type.
- Bilingual labels (*EN/FR*).
- Arrow → Computation block.

⚡ Wizard Page 2: PCB Computation

- Formula: $P=V \cdot I$.
- Optional validation: $V=I \cdot R$.
- Arrow → Relay Control.

🔌 Wizard Page 3: Relay Control

- Threshold check: If PCB value > limit → Relay ON.
- Else → Relay OFF.
- Arrow → Repository Update.

🗄️ Wizard Page 4: Repository Update

- Stores PCB total value, relay state, conductivity.
- Timestamp + bilingual captions.
- Arrow → Audit Log.

📜 Wizard Page 5: Audit Log

- Records Log ID, Timestamp, User Role, Notes.
- Confirms compliance and traceability.
- Arrow → Facilitator Dashboard.

This wizard storyboard makes each module **modular, bilingual, and audit-ready**, while arrows show the **data flow** across the system.

Would you like me to now enrich this storyboard with **icons and schematic overlays** — for example, a voltmeter symbol on Page 1, a formula block on Page 2, a relay switch icon on Page 3, a server stack on Page 4, and a checklist badge on Page 5 — to make it slide-ready for your facilitator deck?

Would you like me to now enrich this storyboard with icons and schematic overlays — for example, a voltmeter symbol on Page 1, a formula block on Page 2, a relay switch icon on Page 3, a server stack on Page 4, and a checklist badge on Page 5 — to make it slide-ready for your facilitator deck?

Yes — enriching your storyboard with **icons and schematic overlays** will make it slide-ready and visually intuitive for facilitators. Here's how each wizard page can be enhanced:

Page 1: Data Reader

- **Voltmeter symbol** → Voltage input.
- **Ammeter symbol** → Current input.
- **Resistor icon** → Resistance input.
- **Material chip icon** → PCB substrate type.

⚡ Page 2: PCB Computation

- **Formula block icon** → P=V·I.
- **Equation overlay** → Ohm's law validation.

Page 3: Relay Control

- **Relay switch symbol** → ON/OFF state.
- **Threshold gauge icon** → Value comparison.

Page 4: Repository Update

- **Server stack icon** → Secure data storage.
- **Database cylinder symbol** → Audit repository.

Page 5: Audit Log

- **Checklist badge icon** → Compliance confirmation.

- **Clock symbol** → Timestamp logging.
- **User profile icon** → Role traceability.

This overlay system makes each wizard page **instantly recognizable** by its iconography, while arrows continue to show the **data flow** across modules. It strengthens bilingual audit readiness and makes your storyboard presentation more engaging.

Would you like me to now design a **consolidated dashboard storyboard** — combining all five wizard pages into one facilitator dashboard with icons, arrows, and bilingual captions?Yes — enriching your storyboard with **icons and schematic overlays** will make it slide-ready and visually intuitive for facilitators. Here's how each wizard page can be enhanced:

🧾 Page 1: Data Reader

- **Voltmeter symbol** → Voltage input.
- **Ammeter symbol** → Current input.
- **Resistor icon** → Resistance input.
- **Material chip icon** → PCB substrate type.

⚡ Page 2: PCB Computation

- **Formula block icon** → P=V·I.
- **Equation overlay** → Ohm's law validation.

🔁 Page 3: Relay Control

- **Relay switch symbol** → ON/OFF state.
- **Threshold gauge icon** → Value comparison.

🗄️ Page 4: Repository Update

- **Server stack icon** → Secure data storage.
- **Database cylinder symbol** → Audit repository.

📜 Page 5: Audit Log

- **Checklist badge icon** → Compliance confirmation.
- **Clock symbol** → Timestamp logging.
- **User profile icon** → Role traceability.

This overlay system makes each wizard page **instantly recognizable** by its iconography, while arrows continue to show the **data flow** across modules. It strengthens bilingual audit readiness and makes your storyboard presentation more engaging.

Would you like me to now design a **consolidated dashboard storyboard** — combining all five wizard pages into one facilitator dashboard with icons, arrows, and bilingual captions?

Would you like me to now design a consolidated dashboard storyboard — combining all five wizard pages into one facilitator dashboard with icons, arrows, and bilingual captions?

Designing a **consolidated dashboard storyboard** will unify all five wizard pages into one facilitator-ready schematic, complete with icons, arrows, and bilingual captions. Here's how the layout would look:

□ Dashboard Integration

- **Top Row / Ligne Supérieure**
  - **Data Reader / Lecteur de Données** → voltmeter, ammeter, resistor, material chip icons.
  - Arrow flows into **PCB Computation / Calcul PCB**.
- **Middle Row / Ligne du Milieu**
  - **PCB Computation / Calcul PCB** → formula block icon (P=V·I).
  - Arrow flows into **Relay Control / Contrôle du Relais** → relay switch icon.
- **Lower Row / Ligne Inférieure**
  - **Repository Update / Mise à Jour Dépôt** → server stack + database cylinder icons.
  - Arrow flows into **Audit Log / Journal d'Audit** → checklist badge + clock + user profile icons.

- **Facilitator Dashboard / Tableau de Bord Facilitateur**
  - Central frame showing bilingual captions for each module.
  - Arrows unify all modules into one compliance-ready system.

✅ Benefits

- **Clarity / Clarté**: All modules visible in one schematic.
- **Traceability / Traçabilité**: Arrows show exact data flow.
- **Audit Readiness / Préparation à l'Audit**: Bilingual captions ensure compliance.
- **Facilitator Utility / Utilité pour Facilitateur**: Easy to present in training or doctoral defense.

This consolidated storyboard makes your VBA wizard system **modular yet unified**, perfect for deployment in your bilingual facilitator deck.

Would you like me to now enrich this dashboard with **color-coded flows** — for example, blue arrows for data, green for repository updates, and red for relay control — to make the schematic even more intuitive?Designing a **consolidated dashboard storyboard** will unify all five wizard pages into one facilitator-ready schematic, complete with icons, arrows, and bilingual captions. Here's how the layout would look:

🧭 Dashboard Integration

- **Top Row / Ligne Supérieure**
  - **Data Reader / Lecteur de Données** → voltmeter, ammeter, resistor, material chip icons.
  - Arrow flows into **PCB Computation / Calcul PCB**.
- **Middle Row / Ligne du Milieu**
  - **PCB Computation / Calcul PCB** → formula block icon (P=V·I).
  - Arrow flows into **Relay Control / Contrôle du Relais** → relay switch icon.
- **Lower Row / Ligne Inférieure**

    - **Repository Update / Mise à Jour Dépôt** → server stack + database cylinder icons.
    - Arrow flows into **Audit Log / Journal d’Audit** → checklist badge + clock + user profile icons.
- **Facilitator Dashboard / Tableau de Bord Facilitateur**
    - Central frame showing bilingual captions for each module.
    - Arrows unify all modules into one compliance-ready system.

✅ Benefits

- **Clarity / Clarté**: All modules visible in one schematic.
- **Traceability / Traçabilité**: Arrows show exact data flow.
- **Audit Readiness / Préparation à l’Audit**: Bilingual captions ensure compliance.
- **Facilitator Utility / Utilité pour Facilitateur**: Easy to present in training or doctoral defense.

This consolidated storyboard makes your VBA wizard system **modular yet unified**, perfect for deployment in your bilingual facilitator deck.

Would you like me to now enrich this dashboard with **color-coded flows** — for example, blue arrows for data, green for repository updates, and red for relay control — to make the schematic even more intuitive?

ould you like me to now enrich this dashboard with color-coded flows — for example, blue arrows for data, green for repository updates, and red for relay control — to make the schematic even more intuitive?

Yes — enriching your consolidated dashboard storyboard with **color-coded flows** will make the schematic much more intuitive and presentation-ready. Here’s how the overlay can be structured:

🎨 Color-Coded Flow Design

- **Blue Arrows / Flèches Bleues → Data Flow**
    - Voltage, Current, Resistance, Material inputs moving into the PCB Computation block.

- **Green Arrows / Flèches Vertes → Repository Updates**
    - Computed PCB values, relay state, and conductivity stored in secure bilingual repository fields.
- **Red Arrows / Flèches Rouges → Relay Control**
    - Threshold triggers activating or deactivating relay switches.
- **Orange Arrows / Flèches Oranges → Audit Log**
    - Timestamp, Log ID, and User Role flowing into compliance records.

✅ Benefits

- **Immediate Recognition / Reconnaissance Immédiate**: Each arrow color signals its function at a glance.
- **Facilitator Clarity / Clarté pour Facilitateur**: Trainers can explain modules without confusion.
- **Audit Readiness / Préparation à l'Audit**: Bilingual captions plus color coding reinforce compliance.
- **System Traceability / Traçabilité du Système**: Data, control, and repository flows are visually distinct.

This enhancement transforms your storyboard into a **slide-ready facilitator dashboard** that is both bilingual and visually intuitive.

Would you like me to now extend this into a **layered storyboard** — where each arrow color overlays the existing schematic, creating a multi-layer view for training and audit presentations?Yes — enriching your consolidated dashboard storyboard with **color-coded flows** will make the schematic much more intuitive and presentation-ready. Here's how the overlay can be structured:

🎨 Color-Coded Flow Design

- **Blue Arrows / Flèches Bleues → Data Flow**
    - Voltage, Current, Resistance, Material inputs moving into the PCB Computation block.

- **Green Arrows / Flèches Vertes → Repository Updates**
    - Computed PCB values, relay state, and conductivity stored in secure bilingual repository fields.
- **Red Arrows / Flèches Rouges → Relay Control**
    - Threshold triggers activating or deactivating relay switches.
- **Orange Arrows / Flèches Oranges → Audit Log**
    - Timestamp, Log ID, and User Role flowing into compliance records.

□ Benefits

- **Immediate Recognition / Reconnaissance Immédiate**: Each arrow color signals its function at a glance.
- **Facilitator Clarity / Clarté pour Facilitateur**: Trainers can explain modules without confusion.
- **Audit Readiness / Préparation à l'Audit**: Bilingual captions plus color coding reinforce compliance.
- **System Traceability / Traçabilité du Système**: Data, control, and repository flows are visually distinct.

This enhancement transforms your storyboard into a **slide-ready facilitator dashboard** that is both bilingual and visually intuitive.

Would you like me to now extend this into a **layered storyboard** — where each arrow color overlays the existing schematic, creating a multi-layer view for training and audit presentations?

Would you like me to now extend this into a layered storyboard — where each arrow color overlays the existing schematic, creating a multi-layer view for training and audit presentations?

Yes — extending into a **layered storyboard** with color-coded overlays will make your facilitator schematic even more powerful. Here's how the multi-layer view can be structured:

□ Layer 1: Data Flow

- **Blue arrows** show sensor inputs (Voltage, Current, Resistance, Material).
- Flow into PCB Computation block.

□ Layer 2: Relay Control

- **Red arrows** highlight threshold checks.
- Relay ON/OFF state visualized with switch icons.

□ Layer 3: Repository Updates

- **Green arrows** connect computation results to repository storage.
- Server stack + database cylinder icons emphasize secure bilingual storage.

□ Layer 4: Audit Integration

- **Orange arrows** trace compliance logging.
- Checklist badge, clock, and user profile icons confirm audit readiness.

□ Benefits of Layered View

- **Clarity / Clarté**: Each arrow color isolates a system function.
- **Training Utility / Utilité pédagogique**: Facilitators can toggle layers to explain modules step by step.
- **Audit Readiness / Préparation à l'audit**: Bilingual captions plus layered flows reinforce compliance.
- **System Traceability / Traçabilité du système**: Multi-layer arrows show how data, control, and compliance interact.

This layered storyboard transforms your dashboard into a **teaching tool and compliance diagram** at once.

Would you like me to now enrich this with **progression bars** — so each module (Data Reader, Computation, Relay, Repository, Audit) shows completion status in bilingual captions for training dashboards?Yes — extending into a **layered storyboard** with color-coded overlays will make

your facilitator schematic even more powerful. Here’s how the multi-layer view can be structured:

🔹 Layer 1: Data Flow

- **Blue arrows** show sensor inputs (Voltage, Current, Resistance, Material).
- Flow into PCB Computation block.

🔹 Layer 2: Relay Control

- **Red arrows** highlight threshold checks.
- Relay ON/OFF state visualized with switch icons.

🔹 Layer 3: Repository Updates

- **Green arrows** connect computation results to repository storage.
- Server stack + database cylinder icons emphasize secure bilingual storage.

🔹 Layer 4: Audit Integration

- **Orange arrows** trace compliance logging.
- Checklist badge, clock, and user profile icons confirm audit readiness.

🔹 Benefits of Layered View

- **Clarity / Clarté**: Each arrow color isolates a system function.
- **Training Utility / Utilité pédagogique**: Facilitators can toggle layers to explain modules step by step.
- **Audit Readiness / Préparation à l’audit**: Bilingual captions plus layered flows reinforce compliance.
- **System Traceability / Traçabilité du système**: Multi-layer arrows show how data, control, and compliance interact.

This layered storyboard transforms your dashboard into a **teaching tool and compliance diagram** at once.

Would you like me to now enrich this with **progression bars** — so each module (Data Reader, Computation, Relay, Repository, Audit) shows completion status in bilingual captions for training dashboards?